

## Large-Scale Simulations of Error-Prone Quantum Computation Devices

Doan Binh Trieu





Forschungszentrum Jülich GmbH  
Institute for Advanced Simulation (IAS)  
Jülich Supercomputing Centre (JSC)

# **Large-Scale Simulations of Error-Prone Quantum Computation Devices**

Doan Binh Trieu

Schriften des Forschungszentrums Jülich  
IAS Series

Volume 2

---

ISSN 1868-8489

ISBN 978-3-89336-601-9

Bibliographic information published by the Deutsche Nationalbibliothek.  
Die Deutsche Nationalbibliothek lists this publication in the  
Deutsche Nationalbibliografie; detailed bibliographic data  
are available in the Internet at <<http://dnb.d-nb.de>>

Publisher  
and Distributor: Forschungszentrum Jülich GmbH  
Zentralbibliothek, Verlag  
D-52425 Jülich  
phone: +49 2461 61-5368 · fax: +49 2461 61-6103  
e-mail: [zb-publikation@fz-juelich.de](mailto:zb-publikation@fz-juelich.de)  
Internet: <http://www.fz-juelich.de/zb>

Cover Design: Grafische Medien, Forschungszentrum Jülich GmbH

Printer: Grafische Medien, Forschungszentrum Jülich GmbH

Copyright: Forschungszentrum Jülich 2009

Schriften des Forschungszentrums Jülich  
IAS Series      Volume 2

D 468 (Diss., Wuppertal, Univ., 2009)

ISSN 1868-8489

ISBN 978-3-89336-601-9

The complete volume is freely available on the Internet on the Jülicher Open Access Server (JUWEL)  
at <http://www.fz-juelich.de/zb/juwel>

Persistent Identifier: urn:nbn:de:0001-00552

Resolving URL: <http://www.persistent-identifier.de/?link=610>

Neither this book nor any part may be reproduced or transmitted in any form or by any means,  
electronic or mechanical, including photocopying, microfilming, and recording, or by any  
information storage and retrieval system, without permission in writing from the publisher.

## Abstract

The theoretical concepts of quantum computation in the idealized and undisturbed case are well understood. However, in practice, all quantum computation devices do suffer from decoherence effects as well as from operational imprecisions.

This work assesses the power of error-prone quantum computation devices using large-scale numerical simulations on parallel supercomputers. We present the *Juelich Massively Parallel Ideal Quantum Computer Simulator (JUMPIQCS)*, that simulates a generic quantum computer on gate level. It comprises an error model for decoherence and operational errors. The robustness of various algorithms in the presence of noise has been analyzed. The simulation results show that for large system sizes and long computations it is imperative to actively correct errors by means of quantum error correction. We implemented the 5-, 7-, and 9-qubit quantum error correction codes. Our simulations confirm that using error-prone correction circuits with non-fault-tolerant quantum error correction will always fail, because more errors are introduced than being corrected. Fault-tolerant methods can overcome this problem, provided that the single qubit error rate is below a certain threshold. We incorporated fault-tolerant quantum error correction techniques into JUMPIQCS using Steane's 7-qubit code and determined this threshold numerically. Using the depolarizing channel as the source of decoherence, we find a threshold error rate of  $(5.2 \pm 0.2) \cdot 10^{-6}$ . For Gaussian distributed operational over-rotations the threshold lies at a standard deviation of  $0.0431 \pm 0.0002$ . We can conclude that quantum error correction is especially well suited for the correction of operational imprecisions and systematic over-rotations.

For realistic simulations of specific quantum computation devices we need to extend the generic model to dynamic simulations, i.e. time-dependent Hamiltonian simulations of realistic hardware models. We focus on today's most advanced technology, i.e. ion trap quantum computation. We developed the *Dynamic Quantum Computer Simulator for Ion Traps (DyQCSI)*. Starting from a microscopic Hamiltonian, it does not rely on approximations that are usually necessary for an analytical approach. We show that the effects due to these approximations are significant. We present several ways for the visualization of the state of the system during its time evolution and demonstrated the benefit of the simulation approach for parameter optimizations.



# Contents

<b>1</b>	<b>Introduction and Overview</b>	<b>1</b>
<b>2</b>	<b>Gate Level Simulations</b>	<b>3</b>
2.1	Ideal Quantum Computer Simulations . . . . .	3
2.1.1	The Circuit Model of Quantum Computation . . . . .	3
2.1.2	Quantum Algorithms . . . . .	8
2.1.2.1	Shor's Algorithm for Prime Factorization . . . . .	8
2.1.2.2	Grover's Search Algorithm . . . . .	14
2.1.3	Massively Parallel Ideal Quantum Computer Simulator . . . . .	18
2.2	Error-Prone Quantum Computer Simulations . . . . .	37
2.2.1	Error Model . . . . .	37
2.2.2	$H^{2k}$ -Algorithm . . . . .	41
2.2.3	Quantum Fourier Transform . . . . .	45
2.2.4	Grover's Search Algorithm . . . . .	50
2.3	Simulation of Quantum Error Correction . . . . .	56
2.3.1	Quantum Error Correction Codes . . . . .	57
2.3.1.1	Shor's 9-Qubit Quantum Error Correction Code . . . . .	58
2.3.1.2	Steane's 7-Qubit Quantum Error Correction Code . . . . .	61
2.3.1.3	5-Qubit Quantum Error Correction Code . . . . .	65
2.3.2	Ideal and Error-Prone Quantum Error Correction . . . . .	68
2.3.3	Fault-Tolerant Quantum Error Correction . . . . .	71
2.3.3.1	Theoretical Principles . . . . .	71
2.3.3.2	Numerical Simulations . . . . .	81
2.3.3.3	Conclusion . . . . .	102



## CONTENTS

---

<b>3</b>	<b>Dynamic Simulations of Ion Trap Quantum Computers</b>	<b>105</b>
3.1	Theory of Ion Trap Quantum Computation . . . . .	105
3.2	Dynamic Quantum Computer Simulator for Ion Traps . . . . .	120
<b>4</b>	<b>Summary and Outlook</b>	<b>139</b>
	<b>Acknowledgements</b>	<b>143</b>
<b>A</b>	<b>Quantum Circuit Symbols</b>	<b>145</b>
<b>B</b>	<b>Equivalence of Depolarizing Channel and Unitary Over-Rotations</b>	<b>147</b>
<b>C</b>	<b>Stabilizer Codes for Quantum Error Correction</b>	<b>153</b>
<b>D</b>	<b>Listings</b>	<b>159</b>

## List of Figures

2.1	Bloch sphere representation of a single qubit . . . . .	4
2.2	Single qubit gates used in quantum computation . . . . .	7
2.3	Controlled-NOT gate . . . . .	7
2.4	Quantum circuit for the quantum Fourier transform . . . . .	9
2.5	Phase estimation procedure . . . . .	10
2.6	Decomposition of Controlled- $U^j$ gate . . . . .	11
2.7	Order-finding algorithm circuit . . . . .	12
2.8	Phase inversions in Grover's algorithm . . . . .	15
2.9	Geometric visualization of a Grover iteration . . . . .	17
2.10	Distribution of state vector to MPI tasks . . . . .	19
2.11	Memory access scheme for a Hadamard operation . . . . .	21
2.12	Pairs of state vector elements . . . . .	22
2.13	Pairs of state vector elements on a distributed memory system . . . . .	22
2.14	Communication scheme in distributed memory . . . . .	23
2.15	Even/odd splitting of state vector . . . . .	24
2.16	Memory access cases for CNOT gate . . . . .	26
2.17	Toffoli gate and decomposition into elementary gates . . . . .	28
2.18	Memory access cases for Toffoli gate . . . . .	30
2.19	Benchmark result of CNOT operation on JUMP (average over control qubits)	32
2.20	Benchmark result of CNOT operation on JUMP (average over target qubits)	33
2.21	Benchmark result of CNOT operation on JUMP: dependency on system size	34
2.22	Benchmark result of CNOT operation on JUMP: dependency on system size and target qubits . . . . .	35

## LIST OF FIGURES

---

2.23	Benchmark result of CNOT operation on JUMP: dependency on control and target qubits . . . . .	36
2.24	Simulation results of $H^{2k}$ -algorithm with depolarizing channel noise . . . .	43
2.25	Simulation results of $H^{2k}$ -algorithm with operational imprecisions . . . . .	44
2.26	Decomposition of swap gate into subsequent CNOT gates . . . . .	45
2.27	Simulation results: error norm for the error-prone quantum Fourier transform	47
2.28	Simulation results: error norm for the quantum Fourier transform under the influence of decoherence errors for various system sizes . . . . .	48
2.29	Visualization of QFT output state . . . . .	48
2.30	Black box oracle for Grover's algorithm . . . . .	50
2.31	$(2 0\rangle\langle 0  - I)$ operator in Grover's algorithm . . . . .	51
2.32	Simulation results of Grover's algorithm under the influence of errors . . . .	52
2.33	Maximum probability of finding the correct database entry with Grover's algorithm in the presence of errors . . . . .	53
2.34	Error norm in Grover's algorithm in the presence of both decoherence and operational errors . . . . .	54
2.35	Visualization of a simulation run of Grover's algorithm . . . . .	54
2.36	Encoding circuit for the 3-qubit bit flip code . . . . .	59
2.37	Syndrome measurement circuit for the 3-qubit bit flip code . . . . .	60
2.38	Encoding circuit for the 3-qubit phase flip code . . . . .	60
2.39	Encoding, recovery and decoding circuit for Shor's 9-qubit code . . . . .	62
2.40	Syndrome measurement circuit for Steane's 7-qubit code . . . . .	64
2.41	Syndrome measurement and recovery in Steane's 7-qubit code . . . . .	64
2.42	Encoding circuit for Steane's 7-qubit code . . . . .	65
2.43	Encoding circuit for the 5-qubit code . . . . .	66
2.44	Syndrome measurement and recovery circuit for the 5-qubit code (non-fault-tolerant) . . . . .	67
2.45	Comparison of 5-, 7-, and 9-qubit quantum error correction codes . . . . .	69
2.46	Results of error-prone quantum error correction (non-fault-tolerant) . . . .	70
2.47	Error propagation through a CNOT gate . . . . .	72
2.48	Concatenation of QEC codes . . . . .	74
2.49	Transversal gates using Steane's 7-qubit code . . . . .	75

2.50	False syndrome extraction . . . . .	76
2.51	Creation of Shor state for fault-tolerant syndrome extraction . . . . .	77
2.52	Fault-tolerant syndrome measurement circuit for Steane's 7-qubit code . . .	79
2.53	CNOT basis transformation . . . . .	79
2.54	Syndrome extraction for phase flip errors using Steane's 7-qubit code . . .	80
2.55	Decoherence on idling qubits . . . . .	83
2.56	Approximate block failure probability . . . . .	85
2.57	Comparison of the fidelities of the unencoded qubit with the qubit protected by Steane's 7-qubit code . . . . .	87
2.58	Gain using fault-tolerant QEC to correct for decoherence errors . . . . .	89
2.59	Relevant parameter range for the determination of the threshold . . . . .	90
2.60	Standard deviation of the mean fidelity while gathering statistics . . . . .	91
2.61	Determination of the threshold and achievable gain for decoherence errors .	92
2.62	Gain from using QEC under the influence of decoherence errors for various algorithm lengths . . . . .	93
2.63	Determination of the threshold and achievable gain for operational errors . .	95
2.64	Determination of the threshold and achievable gain for the combination of decoherence and operational errors . . . . .	96
2.65	Fidelity loss due to systematic over-rotational errors . . . . .	98
2.66	Determination of the threshold and achievable gain for systematic errors . .	99
2.67	Quantum teleportation circuit . . . . .	99
2.68	Relative speedup on JUGENE for a system size of 26 qubits . . . . .	100
3.1	Linear Paul trap . . . . .	106
3.2	Level scheme of $^{40}\text{Ca}^+$ . . . . .	108
3.3	Energy levels for a two-level ion in a harmonic potential . . . . .	111
3.4	Coupling strengths of carrier and sideband transitions . . . . .	112
3.5	Cirac-Zoller CNOT gate . . . . .	114
3.6	Composite pulses CNOT gate . . . . .	115
3.7	Composite pulses controlled phase gate on the Bloch sphere . . . . .	117
3.8	Resonance frequency deviation depending on integration timestep . . . . .	122
3.9	Rabi oscillations on carrier transition and off-resonant . . . . .	124

## LIST OF FIGURES

---

3.10 Rabi oscillations on carrier and sideband . . . . .	125
3.11 Resonance frequencies of the ion trap quantum computer . . . . .	126
3.12 Rabi oscillations for various incident angles . . . . .	127
3.13 Time evolution of the $D_{5/2}$ population during a CNOT gate . . . . .	128
3.14 Spatial probability distribution during a controlled phase gate . . . . .	130
3.15 Bloch sphere representation during a controlled phase gate . . . . .	134

# Chapter 1

## Introduction and Overview

... it seems that the laws of physics present no barrier to reducing the size of computers until bits are the size of atoms, and quantum behavior holds dominant sway.

*(Richard P. Feynman, 1985)*

Since the first ideas of using quantum mechanical systems to store and manipulate information about quantum mechanical systems were formulated [Feynman, 1982; Benioff, 1982], the field of quantum computation and quantum information processing has been evolving rapidly. It has gained several boosts, especially due to the discovery of Shor's prime factorization algorithm [Shor, 1994], which showed the ability of quantum computers to solve computationally hard problems efficiently, and Grover's search algorithm [Grover, 1996], which can search databases faster than classical computers. Quantum computers will most probably be used for the simulation of quantum mechanical systems, which is not possible with classical computers in an efficient manner [Feynman, 1982; Lloyd, 1996; Leibfried et al., 2002; Porras and Cirac, 2004; Byrnes and Yamamoto, 2006]. Quantum computers have actually been realized in the lab based on many different technologies, like nuclear magnetic resonance (NMR) [Gershenfeld and Chuang, 1997], cavity quantum electrodynamics (QED) [Raimond et al., 2001], trapped ions [Monroe et al., 1995; Wineland et al., 1998; Leibfried et al., 2003a] or solid state systems [Loss and DiVincenzo, 1998; Petta et al., 2005]. Although the first proofs of concept have been realized using liquid NMR techniques, very soon it became clear, that such kind of approaches will not be scalable to large systems. However, ion trap quantum computation is a promising candidate technology for large-scale quantum computation [Hughes, 2004], and up to now the largest quantum computer built in the lab comprises eight quantum bits [Häffner et al., 2005]. Critical comments about the power of quantum computers in the presence of noise [Landauer, 1995;

Unruh, 1995] were invalidated with the development of quantum error correction procedures [Shor, 1995; Steane, 1996a,b] and, more importantly, fault-tolerant quantum error correction schemes [Shor, 1996; Aharonov and Ben-Or, 1996; Knill et al., 1996; Preskill, 1998; Gottesman, 1998].

This thesis deals with the issues of error-prone quantum computation devices. A simulation code has been developed, that is named *Juelich Massively Parallel Ideal Quantum Computer Simulator (JUMPIQCS)*. Starting from an error-free ideal quantum computer simulator, which simulates a quantum computer on gate level using the quantum circuit model [Deutsch, 1989], next, an error model is introduced, which includes both imperfect quantum operations as well as interactions with the environment (section 2.2.1). The robustness of different quantum algorithms is analyzed for the given error model.

A practical quantum computer will necessarily incorporate some type of quantum error correction [Preskill, 1998]. After integration of the error model into JUMPIQCS, we extend our analyses to the case of quantum error correction (QEC) (section 2.3). The quantum error correction code involves different correction schemes, i.e. the 5-qubit code, where one logical qubit is encoded into 5 physical ones [Laflamme et al., 1996], Steane’s 7-qubit code [Steane, 1996b] and Shor’s 9-qubit code [Shor, 1995]. The analyses compare theoretically ideal quantum error correction (section 2.3.1) with error-prone quantum error correction (section 2.3.2), where the correction steps themselves are also prone to error. Our results reinforce that straightforward ideal quantum error correction schemes break down if the correction circuit is also prone to error. Thus, fault-tolerant quantum error correction schemes are mandatory for a successful application of quantum error correction (section 2.3.3). Fault-tolerant quantum error correction requires the single qubit gate error probability to be below a certain threshold. We determine this threshold numerically.

For realistic simulations of specific quantum computation devices, one needs to extend the gate model to dynamic simulations, i.e. time-dependent Hamiltonian simulations of realistic hardware models. We focus on today’s most advanced technology, i.e. ion trap quantum computation (chapter 3), and simulate this system by solving the time-dependent Schrödinger equation for a chain of trapped ions coupled by a common vibrational phonon mode. The simulation package *Dynamic Quantum Computer Simulator for Ion Traps (DyQCSI)* involves all the basic operations needed for the simulation of an ion trap quantum computer. In a first stage, the model Hamiltonian will be fine-tuned to the experimental situation and, in the future, one can use the simulation as a tool to guide the optimization of the error-prone experimental ion trap quantum computer.

## Chapter 2

### Gate Level Simulations

Nowadays, classical computers work with a well defined computational model, namely manipulation of bits by means of Boolean logic. By contrast, in quantum computation there are several different models of computation, like adiabatic quantum computation [Farhi et al., 2000], cluster state quantum computation [Raussendorf and Briegel, 2001] or topological quantum computation [Kitaev, 1997b]. The most common model of quantum computation today is the quantum circuit model [Deutsch, 1989]. In analogy to classical circuits where bits and logic gates are used to form a circuit, a quantum computer circuit consists of quantum bits (qubits) and quantum gates, which operate on those qubits.

This chapter of the thesis stays within the framework of the quantum circuit model. The first section deals with idealized quantum computation, i.e., all gate operations are assumed to be error-free. It is followed by a section where analyses are made for the case of non-ideal, i.e. error-prone gate operations. After examining the behavior of quantum algorithms under the influence of errors in the second section, the third section explains how quantum algorithms, that are prone to error, can be actively improved by using quantum error correction.

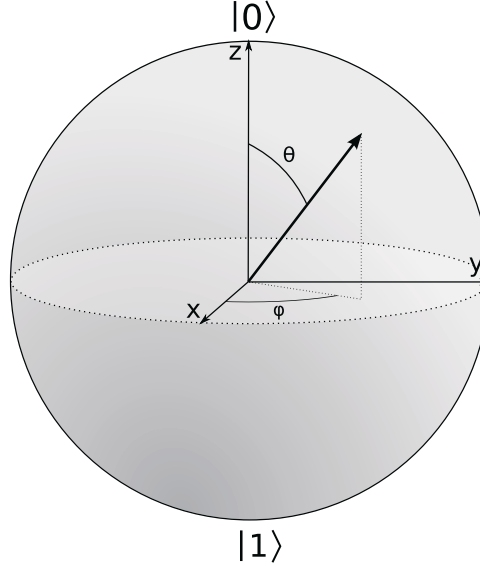
#### 2.1 Ideal Quantum Computer Simulations

We will briefly describe the basics of quantum computation in the quantum circuit model and the two most important quantum algorithms, namely Shor's algorithm [Shor, 1994] and Grover's algorithm [Grover, 1996]. These are the algorithms that are examined with the *Juelich Massively Parallel Ideal Quantum Computer Simulator (JUMPIQCS)*.

##### 2.1.1 The Circuit Model of Quantum Computation

**The quantum bit** In contrast to a classical bit, which can only be in the state 0 or 1, a quantum bit consists of a quantum-mechanical two-level system with basis states  $|0\rangle$  and





**Figure 2.1** – Bloch sphere representation of a single qubit. The state of a single qubit can be represented by a vector on the Bloch sphere. The state  $|0\rangle$  can be chosen to point to the top,  $|1\rangle$  then points down. A qubit can be in an arbitrary superposition of these states, each represented by a point on the surface of this unit sphere. Mixed states are represented by vectors of length smaller than one within the Bloch sphere.

$|1\rangle$ . According to the laws of quantum mechanics the quantum bit can be in a superposition of both states,

$$|\psi\rangle = \alpha |0\rangle + \beta |1\rangle \cong \begin{pmatrix} \alpha \\ \beta \end{pmatrix}, \quad (2.1)$$

with

$$\alpha, \beta \in \mathbb{C}, \quad |\alpha|^2 + |\beta|^2 = 1. \quad (2.2)$$

Every single qubit can be represented by a vector on the Bloch sphere. Because of the normalization equation (2.2), the qubit can be written as

$$|\psi\rangle = \cos \frac{\theta}{2} |0\rangle + e^{i\varphi} \sin \frac{\theta}{2} |1\rangle, \quad (2.3)$$

where  $\theta$  and  $\varphi$  are real numbers and therefore define a point on the Bloch sphere (see figure 2.1). A global phase factor can be omitted, so that the  $|0\rangle$ -axis can be chosen to be real.

**Multiple quantum bits and entanglement** Multiple quantum bits, say  $n$  qubits, can be described by a  $2^n$ -dimensional complex vector, e.g., the state of two qubits is described by

$$|\psi\rangle = \alpha_{00}|00\rangle + \alpha_{01}|01\rangle + \alpha_{10}|10\rangle + \alpha_{11}|11\rangle \hat{=} \begin{pmatrix} \alpha_{00} \\ \alpha_{01} \\ \alpha_{10} \\ \alpha_{11} \end{pmatrix}, \quad (2.4)$$

$$\alpha_{ij} \in \mathbb{C}, \quad \sum |\alpha_{ij}|^2 = 1.$$

Unlike in classical computation, multiple qubits can be in an *entangled state* [Schrödinger, 1935], which exhibits unique quantum correlations between its constituents. The overall state cannot be described by specifying the state of the individual qubits separately. For example, the two qubit system

$$|\psi\rangle = \frac{1}{\sqrt{2}}(|00\rangle + |11\rangle) \quad (2.5)$$

is a maximally entangled state. It does not have a representation as a product of independent single qubit states. For arbitrary qubits

$$|\psi_1\rangle = \alpha_1|0\rangle + \beta_1|1\rangle \quad (2.6)$$

and

$$|\psi_2\rangle = \alpha_2|0\rangle + \beta_2|1\rangle, \quad (2.7)$$

with  $\alpha_1, \beta_1, \alpha_2, \beta_2 \in \mathbb{C}$ , the product gives

$$|\psi_1\rangle \otimes |\psi_2\rangle = \alpha_1\alpha_2|00\rangle + \alpha_1\beta_2|01\rangle + \alpha_2\beta_1|10\rangle + \beta_1\beta_2|11\rangle, \quad (2.8)$$

and it is impossible to choose the coefficients to give the entangled state  $|\psi\rangle$  (equation (2.5)). This entangled state exhibits a perfect correlation between its individual components. If one qubit is measured, the measurement of the second one will always give the same result. These quantum correlations are exploited directly in quantum communication processes and can be regarded as a resource for quantum computation.

**Quantum computation** The behavior of a quantum mechanical system is governed by the Schrödinger equation

$$i\hbar \frac{\partial}{\partial t} |\psi(t)\rangle = H(t) |\psi(t)\rangle, \quad (2.9)$$

where  $|\psi\rangle$  is the wave function describing the state of the system,  $t$  is the time variable,  $H$  is the hermitian Hamiltonian of the system, and  $\hbar$  is Planck's constant.

From the normalization condition

$$\langle\psi|\psi\rangle = 1 \quad (2.10)$$

it follows immediately that quantum operations are defined by reversible unitary operators. The time evolution of the system is given by the time evolution operator

$$U(t_2, t_1) = T e^{-\frac{i}{\hbar} \int_{t_1}^{t_2} H(t) dt}, \quad (2.11)$$

where the time order operator  $T$  is needed if the commutator  $[H(t'), H(t'')] \neq 0$ .

Thus, a quantum computer takes some complex vector and computes the output, another complex vector of the same dimension, by applying a linear unitary operator  $U$ . In principle, this is just a complex-valued matrix-vector multiplication:

$$\begin{pmatrix} \alpha'_1 \\ \alpha'_2 \\ \vdots \\ \alpha'_{2^n} \end{pmatrix} = \begin{pmatrix} u_{11} & u_{12} & \dots & u_{12^n} \\ u_{21} & u_{22} & \dots & u_{22^n} \\ \vdots & \vdots & \ddots & \vdots \\ u_{2^n 1} & u_{2^n 2} & \dots & u_{2^n 2^n} \end{pmatrix} \begin{pmatrix} \alpha_1 \\ \alpha_2 \\ \vdots \\ \alpha_{2^n} \end{pmatrix}. \quad (2.12)$$

The size of the state vector as well as the unitary matrix describing the time evolution grow exponentially with the number of qubits.

In the circuit model of quantum computation the evolution is decomposed into a sequence of unitary operators acting only on a subspace of the whole Hilbert space. These operators are called quantum gates and an  $n$ -qubit gate is represented by a  $2^n \times 2^n$  unitary matrix.

The graphical representation of a quantum circuit uses well defined schematic symbols. It uses wires and gates to depict the time evolution of the quantum system. Here, each qubit is represented by a horizontal line going from left to right indicating the time evolution, i.e., the complete circuit is loop free. Vertical lines connect different qubits of the system. A bullet specifies a qubit that controls an operation on a connected qubit. See appendix A for a complete list of symbols used.

The most common 1-qubit gates in quantum computation are shown in figure 2.2. The Pauli matrices are the generators of rotations on the Bloch sphere, while the Hadamard gate turns a well defined qubit into a superposition state. The  $T$ -gate, also called  $\pi/8$ -gate, can be used as a basic gate in a set of gates that can provide universal quantum computation. The  $S$ -gate or phase-gate can actually be constructed from two  $T$ -gates.

Interactions between qubits are realized with gates which act on at least two qubits. The most commonly used one is the controlled-NOT (CNOT) gate (see figure 2.3), which is the quantum analogue to the classical XOR gate. The CNOT gate flips the state of the target qubit conditioned on the state of the control qubit.

**Universal Quantum Computation** It has been proven, that a universal quantum computer can simulate any Turing machine [Deutsch, 1985] and any local quantum system [Lloyd, 1996]. A set of gates is called universal if any unitary operation can be approximated to arbitrary accuracy by a quantum circuit consisting of those gates only.

$$\begin{array}{lll}
 \text{Pauli-X} & \text{---} \boxed{X} \text{---} & \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \\
 \text{Pauli-Y} & \text{---} \boxed{Y} \text{---} & \begin{pmatrix} 0 & -i \\ i & 0 \end{pmatrix} \\
 \text{Pauli-Z} & \text{---} \boxed{Z} \text{---} & \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix} \\
 \text{Phase} & \text{---} \boxed{S} \text{---} & \begin{pmatrix} 1 & 0 \\ 0 & i \end{pmatrix} \\
 \pi/8 & \text{---} \boxed{T} \text{---} & \begin{pmatrix} 1 & 0 \\ 0 & e^{i\pi/4} \end{pmatrix} \\
 \text{Hadamard} & \text{---} \boxed{H} \text{---} & \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}
 \end{array}$$

**Figure 2.2** – Single qubit gates used in quantum computation. The Pauli-X gate corresponds to the classical NOT gate.

$$\text{CNOT} \quad \begin{array}{c} |c\rangle \text{---} \bullet \text{---} |c\rangle \\ |t\rangle \text{---} \oplus \text{---} |t \oplus c\rangle \end{array} \quad \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix}$$

**Figure 2.3** – The controlled-NOT (CNOT) gate is the quantum generalization of the classical XOR gate. It flips the target qubit  $t$  conditioned on the control qubit  $c$ . It does the transformation  $|c\rangle |t\rangle \rightarrow |c\rangle |t \oplus c\rangle$ , where  $\oplus$  means addition modulo two. The effect on the computational basis states is:  $|00\rangle \rightarrow |00\rangle$ ,  $|01\rangle \rightarrow |01\rangle$ ,  $|10\rangle \rightarrow |11\rangle$ , and  $|11\rangle \rightarrow |10\rangle$ .

It has been shown ([DiVincenzo, 1995]) that any unitary operation can be approximated to arbitrary accuracy by using only single qubit gates and the CNOT gate. Moreover, it has been proven that a discrete set of gates, e.g. the Hadamard, phase, CNOT and  $\pi/8$ -gate, is sufficient for universal quantum computation [Barenco et al., 1995]. Another possible universal set of quantum gates consists of the Hadamard, phase and Toffoli gate<sup>1</sup> [Nielsen and Chuang, 2000]. However, there are cases where the overhead or the amount of gates used for this approximation grows exponentially. The aim of quantum computation in the quantum circuit model is to take advantage of algorithms that can be performed efficiently. The restriction to a discrete set will be exploited for fault-tolerant quantum computation (section 2.3), where it is sufficient to construct this set of gates fault-tolerantly using quantum error correction codes.

<sup>1</sup>The Toffoli gate, a three qubit gate, can be regarded as generalized CNOT gate with two control bits and one target bit. The target bit is flipped if and only if both control bits are set to one (see appendix A).

## 2.1.2 Quantum Algorithms

### 2.1.2.1 Shor's Algorithm for Prime Factorization

Due to Shor's algorithm for prime factorization [Shor, 1994] the research area of quantum information gained a significant boost, since this algorithm provides evidence, that quantum computers are able to solve tasks efficiently, that are considered intractable on classical computers. This has great impact on the reliability of today's encryption schemes, because they mostly rely on the hardness of prime factorization. Prime factorization is a problem from the complexity class **NP**, i.e. Non-deterministic Polynomial time. This is the class of problems for which solutions are hard to find, but easy to verify. Since there is no known classical algorithm for prime factorization which works in polynomial time, Shor's algorithm gives an exponential speedup. Quantum computers are considered to be in principle inherently more powerful than classical ones.

Factoring can be regarded as an application of the order-finding algorithm, also known as period-finding, because it searches for the period of a modular exponential function. Order-finding can be done efficiently by using the phase estimation procedure, which relies on the quantum Fourier transform (QFT). We will explain Shor's algorithm for prime factorization, by describing its kernel, i.e. the quantum Fourier transform, then how this is used for doing a phase estimation and how that can be applied to accomplish order-finding. In the last step we briefly describe how factoring can be reduced to order-finding. The following is a brief summary of a more detailed description given in [Nielsen and Chuang, 2000].

**Quantum Fourier transform** The core routine of Shor's algorithm is the quantum Fourier transform, which performs the Fourier transform of quantum mechanical amplitudes. In analogy to the classical discrete Fourier transform

$$y_k = \frac{1}{\sqrt{N}} \sum_{j=0}^{N-1} x_j e^{2\pi i j k / N}, \quad (2.13)$$

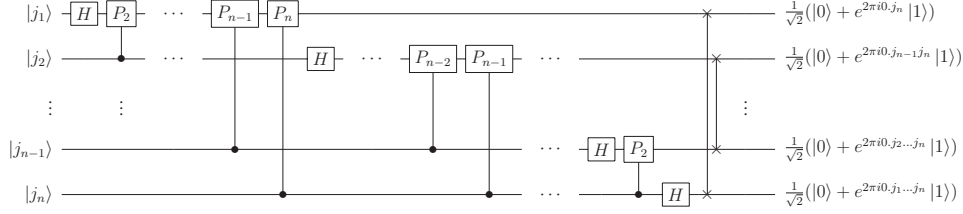
which transforms an input vector of complex numbers,  $x_0, \dots, x_{N-1}$  to an output vector  $y_0, \dots, y_{N-1}$ , the quantum Fourier transform does exactly the same for an orthonormal basis  $|0\rangle, \dots, |N-1\rangle$ , i.e.,

$$|j\rangle \rightarrow \frac{1}{\sqrt{N}} \sum_{k=0}^{N-1} e^{2\pi i j k / N} |k\rangle. \quad (2.14)$$

An arbitrary state is transformed as

$$\sum_{j=0}^{N-1} x_j |j\rangle \rightarrow \sum_{k=0}^{N-1} y_k |k\rangle, \quad (2.15)$$

## 2.1. IDEAL QUANTUM COMPUTER SIMULATIONS



**Figure 2.4** – Quantum circuit for the quantum Fourier transform. The circuit can be derived from the product representation of the quantum Fourier transform (equation (2.16)). The operations  $P_k$  are phase rotations defined in equation (2.17). Swap gates (see appendix A) at the end of the circuit reverse the order of the qubits.

with  $y_k$  being the discrete Fourier transforms of the amplitudes  $x_j$ .

Using the binary representation  $j = j_1 2^{n-1} + j_2 2^{n-2} + \dots + j_n 2^0 = j_1 j_2 \dots j_n$  and the notation for the binary fraction  $j_l/2 + j_{l+1}/4 + \dots + j_m/2^{m-l+1} = 0.j_l j_{l+1} \dots j_m$ , the quantum Fourier transform can be written in product representation [Nielsen and Chuang, 2000]:

$$|j_1 \dots j_n\rangle \rightarrow \frac{(|0\rangle + e^{2\pi i 0 \cdot j_n} |1\rangle)(|0\rangle + e^{2\pi i 0 \cdot j_{n-1} j_n} |1\rangle) \dots (|0\rangle + e^{2\pi i 0 \cdot j_1 j_2 \dots j_n} |1\rangle)}{2^{n/2}}. \quad (2.16)$$

Using the definition of phase rotations

$$P_k = \begin{pmatrix} 1 & 0 \\ 0 & e^{2\pi i / 2^k} \end{pmatrix}, \quad (2.17)$$

the product representation leads to the construction scheme for the QFT circuit (see figure 2.4).

Starting with  $|j_1 \dots j_n\rangle$  and applying a Hadamard gate on the first qubits gives

$$\frac{1}{2^{1/2}} (|0\rangle + e^{2\pi i 0 \cdot j_1} |1\rangle) |j_2 \dots j_n\rangle. \quad (2.18)$$

The controlled- $P_2$  leads to

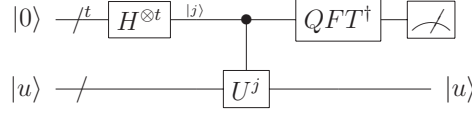
$$\frac{1}{2^{1/2}} (|0\rangle + e^{2\pi i 0 \cdot j_1 j_2} |1\rangle) |j_2 \dots j_n\rangle \quad (2.19)$$

and subsequent controlled- $P_k$  gates produce the state

$$\frac{1}{2^{1/2}} (|0\rangle + e^{2\pi i 0 \cdot j_1 j_2 \dots j_n} |1\rangle) |j_2 \dots j_n\rangle. \quad (2.20)$$

The procedure on the second qubit then leads to

$$\frac{1}{2^{2/2}} (|0\rangle + e^{2\pi i 0 \cdot j_1 j_2 \dots j_n} |1\rangle) (|0\rangle + e^{2\pi i 0 \cdot j_2 \dots j_n} |1\rangle) |j_3 \dots j_n\rangle \quad (2.21)$$



**Figure 2.5** – Overview of the phase estimation procedure. The first register consists of  $t$  qubits to represent the phase  $\phi$ . The second register holds the eigenstate  $|u\rangle$ . The step before measurement is an inverse quantum Fourier transform. The controlled- $U^j$  gate takes the state  $|j\rangle |u\rangle$  to  $|j\rangle U^j |u\rangle$  and can also be represented by the decomposition shown in figure 2.6.

and on the remaining qubits finally gives

$$\frac{1}{2^{n/2}} \left( |0\rangle + e^{2\pi i 0.j_1 j_2 \dots j_n} |1\rangle \right) \left( |0\rangle + e^{2\pi i 0.j_2 \dots j_n} |1\rangle \right) \dots \left( |0\rangle + e^{2\pi i 0.j_n} |1\rangle \right). \quad (2.22)$$

After reversal of the qubits, indicated by the swap gates (see appendix A) at the end of the circuit, the state is the Fourier transform of the initial state (equation (2.16)).

A look at the resource requirement, with  $N = 2^n$ , shows that the circuit needs  $n + (n - 1) + \dots + 1 = n(n + 1)/2$  gates plus  $n/2$  additional swap gates (which can be build from three CNOT gates as shown in figure 2.26). In total this is a  $\mathcal{O}(n^2)$  algorithm for performing the Fourier transform. In contrast to that, the best known classical algorithm for the Fourier transform, the fast Fourier transform (FFT) needs  $\mathcal{O}(n2^n)$  gates to compute the Fourier transform of  $2^n$  elements. Although it seems as if the QFT provides an exponential speedup over the FFT for doing a discrete Fourier transform, this is not the case, because there is no way to directly access all of the transformed amplitudes by measurement. There are more sophisticated ways to exploit the power of the QFT. This will be explained in the following paragraph.

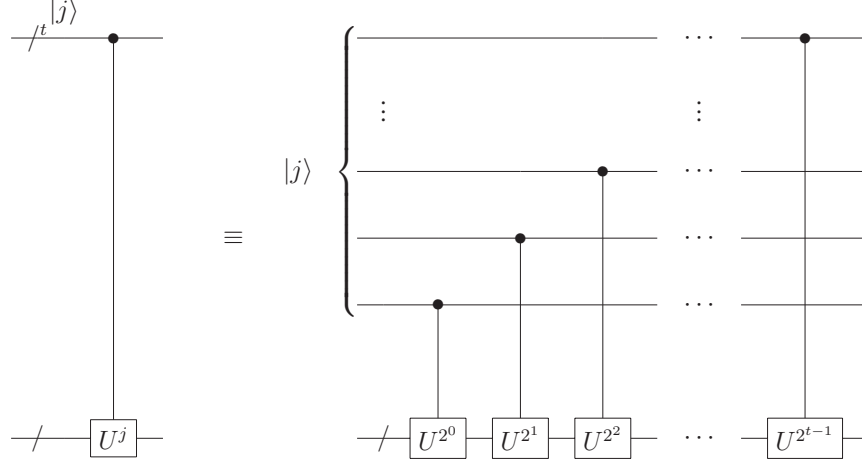
**Phase estimation** The QFT can be used for phase estimation. Suppose there is a given unitary operator  $U$  with an eigenstate  $|u\rangle$  and eigenvalue  $e^{2\pi i \phi}$ . Then the phase  $\phi$  can be determined efficiently by the phase estimation algorithm schematically depicted in figure 2.5.

Two registers are needed, the first holding  $t$  qubits to represent the phase  $\phi$ , the second containing the eigenstate  $|u\rangle$ . After applying the controlled- $U^j$  gate shown in figures 2.5 and 2.6 the first register is in the state

$$\begin{aligned} \frac{1}{2^{t/2}} \left( |0\rangle + e^{2\pi i 2^{t-1} \phi} |1\rangle \right) \left( |0\rangle + e^{2\pi i 2^{t-2} \phi} |1\rangle \right) \dots \left( |0\rangle + e^{2\pi i 2^0 \phi} |1\rangle \right) \\ = \frac{1}{2^{t/2}} \sum_{k=0}^{2^t-1} e^{2\pi i \phi k} |k\rangle. \end{aligned} \quad (2.23)$$

If  $\phi$  can be expressed exactly in  $t$  bits, i.e.  $\phi = 0.\phi_1 \dots \phi_t$ , equation (2.23) can be written as

$$\frac{1}{2^{t/2}} \left( |0\rangle + e^{2\pi i 0.\phi_t} |1\rangle \right) \left( |0\rangle + e^{2\pi i 0.\phi_{t-1} \phi_t} |1\rangle \right) \dots \left( |0\rangle + e^{2\pi i 0.\phi_1 \phi_2 \dots \phi_t} |1\rangle \right). \quad (2.24)$$



**Figure 2.6** – The controlled- $U^j$  gate used during the phase estimation algorithm (figure 2.5) can be decomposed into a sequence of controlled gates of  $U$  raised to successive powers of two. If the positive integer  $j$  is represented by  $t$  bits, i.e.  $j = \sum_{k=0}^{t-1} j_k 2^k$ ,  $U^j$  can be written as  $U^j = U^{j_{t-1} 2^{t-1}} \dots U^{j_0 2^0}$ .

After application of the inverse Fourier transform (see equation (2.16)) the final state of the first register will be  $|\phi_1 \dots \phi_t\rangle$  exactly.

The generalization to the case where the phase  $\phi$  cannot be exactly expressed with a  $t$  bit binary expansion is covered in detail in [Nielsen and Chuang, 2000]. In that case the estimated phase will be a good approximation to the exact phase.

**Order-finding** The order-finding problem is defined as the problem of finding the smallest positive integer number  $r$  such that

$$x^r = 1 \pmod{N}, \quad (2.25)$$

for positive integers  $x$  and  $N$ , that have no common factors and where  $x < N$ .

The order-finding problem can be solved by applying the phase estimation algorithm to the unitary transformation

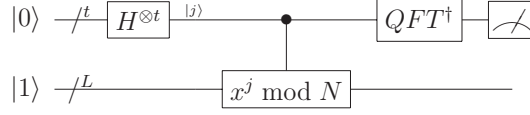
$$U|y\rangle = |xy \pmod{N}\rangle, \quad (2.26)$$

with  $y \in \{0, 1\}^L$  and  $L \equiv \lceil \log(N) \rceil$  being the number of bits needed to specify  $N$ .

The eigenstates we are interested in are

$$|u_s\rangle = \frac{1}{\sqrt{r}} \sum_{k=0}^{r-1} e^{\frac{-2\pi i s k}{r}} |x^k \pmod{N}\rangle, \quad (2.27)$$





**Figure 2.7** – Overview of the order finding algorithm.

with integer  $s$ ,  $0 \leq s \leq r - 1$ , and

$$U |u_s\rangle = e^{\frac{2\pi i s}{r}} |u_s\rangle. \quad (2.28)$$

The phase estimation algorithm determines the eigenvalue  $e^{2\pi i s/r}$  from which the order  $r$  can be derived.

Two requirements have to be met: An efficient implementation of the controlled- $U^{2^j}$  operations for any integer  $j$  and the efficient preparation of the eigenstate  $|u_s\rangle$ . The controlled- $U^j$  operation can be efficiently performed by using the modular exponentiation technique using  $\mathcal{O}(L^3)$  gates (see [Nielsen and Chuang, 2000]). The preparation of  $|u_s\rangle$  seems to require  $r$  to be known in advance, but considering the fact that (see [Nielsen and Chuang, 2000])

$$\frac{1}{\sqrt{r}} \sum_{s=0}^{r-1} |u_s\rangle = |1\rangle, \quad (2.29)$$

we can prepare two registers, the first one (with  $t = 2L + 1 + \lceil \log(2 + 1/(2\epsilon)) \rceil$  qubits) in the state  $|0\rangle$  and the second one in  $|1\rangle$ , so that for each  $s$ ,  $0 \leq s \leq r - 1$ , the phase  $\phi \approx s/r$  will be estimated with an accuracy of  $2L + 1$  bits and probability greater than  $(1 - \epsilon)/r$  (see figure 2.7). From the phase  $\phi \approx s/r$  the order  $r$  can be efficiently ( $\mathcal{O}(L^3)$ ) derived by using the continued fraction expansion, that describes real numbers in terms of integers [Nielsen and Chuang, 2000]:

$$[a_0, \dots, a_M] = a_0 + \frac{1}{a_1 + \frac{1}{a_2 + \frac{1}{\dots + \frac{1}{a_M}}}} \quad (2.30)$$

For rational numbers this algorithm terminates after a finite number of steps, otherwise  $[a_0, \dots, a_m]$ ,  $0 \leq m \leq M$  is defined as the  $m^{\text{th}}$  convergent.

**Factoring** Factoring is one application of the order-finding algorithm. The procedure goes as follows: One starts with a composite number  $N$ . If  $N$  is even, 2 is a trivial factor. If  $N = a^b$ , with  $a \geq 1$  and  $b \geq 2$ ,  $a$  is a factor. Otherwise choose a random  $x$ ,  $1 \leq x \leq N - 1$  and calculate  $\gcd(x, N)$ . If this is greater than one, we have found a factor. So far the algorithm is purely classical. Now, use the order-finding subroutine to find the order  $r$  of  $x \bmod N$  (equation (2.25)). This is the quantum part of the algorithm. From number theoretical considerations it follows that if  $r$  is even and  $x^{r/2} \not\equiv -1 \pmod{N}$  (the probability for that is guaranteed to be at least  $1/2$ ),  $\gcd(x^{r/2} - 1, N)$  and  $\gcd(x^{r/2} + 1, N)$

are possible candidates for being a non-trivial factor. This can be easily checked, since factoring is an **NP**-problem. If this step does not give a non-trivial factor, the algorithm fails and one has to restart. The final result is a non-trivial factor of  $N$  with success probability of  $\mathcal{O}(1)$  using  $\mathcal{O}((\log N)^3)$  operations.

### 2.1.2.2 Grover's Search Algorithm

Grover's algorithm [Grover, 1996] exploits quantum mechanics to carry out a fast database search. Searching through a database with  $N$  elements requires only  $\mathcal{O}(\sqrt{N})$  operations, whereas in the classical case one needs  $\mathcal{O}(N)$  operations.

This section is based on the description given in [Nielsen and Chuang, 2000]. Given a search space of  $N = 2^n$  elements, we search through the indices from 0 to  $N - 1$ , which can be stored in  $n$  qubits. We assume that the search problem has  $M$  solutions,  $1 \leq M \leq N$ . These solutions are described by a function  $f$ :

$$f(x) = \begin{cases} 1 & \text{if } x \text{ is a solution} \\ 0 & \text{if } x \text{ is not a solution.} \end{cases} \quad (2.31)$$

Grover's algorithm works with a unitary operator  $O$  called the oracle function, which is defined by

$$|x\rangle |q\rangle \xrightarrow{O} |x\rangle |q \oplus f(x)\rangle, \quad (2.32)$$

where  $|x\rangle$  is the  $n$  qubit index register and  $|q\rangle$  is an additional qubit, called the oracle qubit.

The first step is to prepare the first register in a uniform superposition by applying  $H^{\otimes n}$  to  $|0\rangle^{\otimes n}$  and the oracle qubit in  $2^{-1/2}(|0\rangle - |1\rangle)$  by applying  $HX$  to  $|0\rangle$ , so that we get the state

$$\frac{1}{\sqrt{2^n}} \sum_{x=0}^{2^n-1} |x\rangle \left( \frac{|0\rangle - |1\rangle}{\sqrt{2}} \right). \quad (2.33)$$

The oracle marks elements of the solution by inverting their phases (see figure 2.8),

$$|x\rangle \xrightarrow{O} (-1)^{f(x)} |x\rangle. \quad (2.34)$$

The oracle qubit remains in the state  $2^{-1/2}(|0\rangle - |1\rangle)$  throughout the rest of the algorithm.

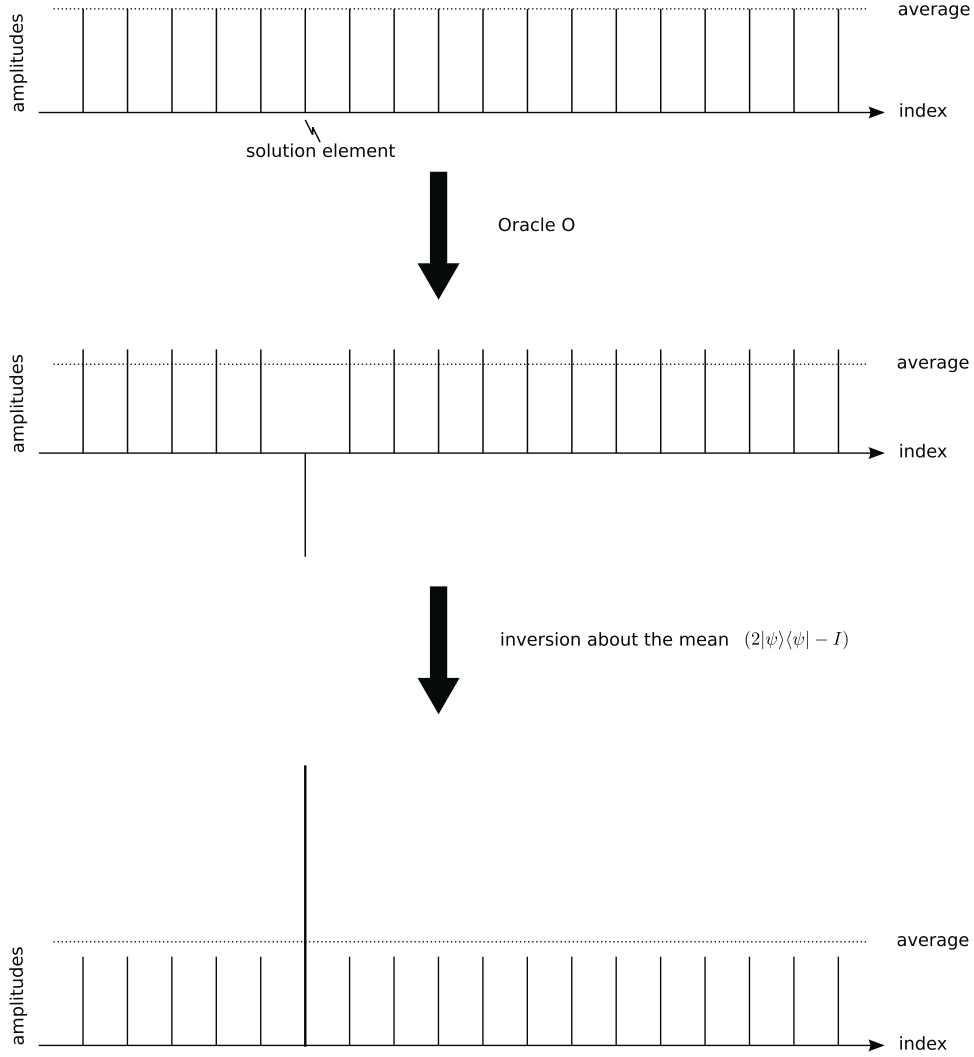
Grover's algorithm consists of repeated applications of Grover iterations  $G$  defined as

$$G = H^{\otimes n} \left( 2|0\rangle\langle 0| - I \right) H^{\otimes n} O = \left( 2|\psi\rangle\langle\psi| - I \right) O, \quad (2.35)$$

where  $|\psi\rangle = 2^{-n/2} \sum_{x=0}^{2^n-1} |x\rangle$  is the state of uniform superposition. The Grover iteration  $G$  is an oracle call followed by the operator  $(2|\psi\rangle\langle\psi| - I)$ . The oracle call flips the phase of the amplitudes of the solution elements and the subsequent operator does an inversion about the mean (see figure 2.8).

**Geometric visualization of the Grover iteration** The Grover iteration step  $G = (2|\psi\rangle\langle\psi| - I) O$  can be regarded as a two-dimensional rotation in the plane spanned

## 2.1. IDEAL QUANTUM COMPUTER SIMULATIONS



**Figure 2.8** – Grover’s algorithm: Phase inversion of the oracle and subsequent inversion about the mean. In this example there is only a single solution element.

by the starting vector of uniform superposition  $|\psi\rangle$  and the vector of equal superposition of solution elements.

We define the set of solutions  $S = \{x | f(x) = 1\}$  and the normalized states

$$|\alpha\rangle = \frac{1}{\sqrt{N-M}} \sum_{x \notin S} |x\rangle \quad (2.36)$$

and

$$|\beta\rangle = \frac{1}{\sqrt{M}} \sum_{x \in S} |x\rangle, \quad (2.37)$$

so that

$$|\psi\rangle = \sqrt{\frac{N-M}{N}} |\alpha\rangle + \sqrt{\frac{M}{N}} |\beta\rangle. \quad (2.38)$$

The effect of the oracle  $O$  is to reflect the start vector about the vector  $|\alpha\rangle$  in the plane spanned by  $|\alpha\rangle$  and  $|\beta\rangle$ , whereas  $(2|\psi\rangle\langle\psi| - I)$  is a reflection about  $|\psi\rangle$ . Altogether, a Grover iteration rotates the state in the space spanned by  $|\alpha\rangle$  and  $|\beta\rangle$ . With the definition

$$\cos \frac{\theta}{2} = \sqrt{\frac{N-M}{N}}, \quad (2.39)$$

$|\psi\rangle$  can be expressed as

$$|\psi\rangle = \cos \frac{\theta}{2} |\alpha\rangle + \sin \frac{\theta}{2} |\beta\rangle, \quad (2.40)$$

and

$$G|\psi\rangle = \cos \frac{3\theta}{2} |\alpha\rangle + \sin \frac{3\theta}{2} |\beta\rangle. \quad (2.41)$$

Repeated application of the Grover iteration leads to

$$G^k |\psi\rangle = \cos \left( \frac{2k+1}{2} \theta \right) |\alpha\rangle + \sin \left( \frac{2k+1}{2} \theta \right) |\beta\rangle. \quad (2.42)$$

This means that the start vector is rotated towards the space of the solutions  $|\beta\rangle$  (see figure 2.9). The essence is to rotate  $|\psi\rangle$  close to  $|\beta\rangle$  by doing a well defined number of Grover iterations. According to equation (2.38), a rotation about the angle  $\arccos((M/N)^{1/2})$  takes the vector  $|\psi\rangle$  to the state  $|\beta\rangle$ . That means, repeating the Grover iteration  $R$  times, with

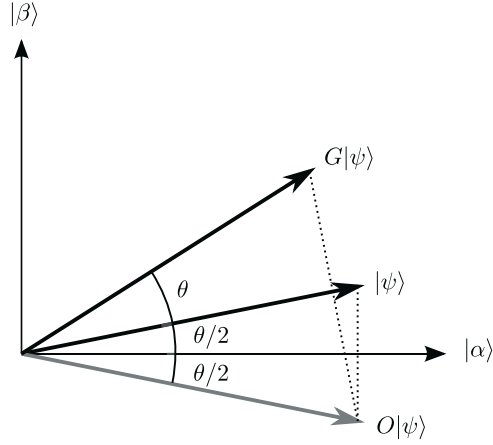
$$R = \left\lceil \left( \frac{\arccos \sqrt{\frac{M}{N}}}{\theta} \right) + 0.5 \right\rceil, \quad (2.43)$$

takes the system close to  $|\beta\rangle$  with a deviation of at most  $\theta/2$ .

For  $M \ll N$  we have  $\theta \approx \sin \theta \approx 2\sqrt{M/N}$  and the error is at most  $\theta/2 \approx \sqrt{M/N}$ , so that the largest possible probability of error is  $M/N$ .

An upper bound for the number of necessary iterations can be derived from the lower bound on  $\theta$  (see equation (2.43)), i.e.  $R \leq \lceil \pi/(2\theta) \rceil$ , so that (assuming  $M \leq N/2$ )

$$\frac{\theta}{2} \geq \sin \frac{\theta}{2} = \sqrt{\frac{M}{N}}. \quad (2.44)$$



**Figure 2.9** – Geometric visualization of a Grover iteration. Starting from the state of uniform superposition  $|\psi\rangle$ , the oracle operation  $O$  reflects this state about the equal superposition of non-solutions  $|\alpha\rangle$ , followed by a reflection about  $|\psi\rangle$ . That is, a Grover iteration is equal to a rotation by  $\theta$ . Repeated application of the Grover iteration rotates the state towards the space of solutions  $|\beta\rangle$ . One has to make sure to do a well defined number of iterations, so that the vector stops close to the vector of solutions. Figure modified from [Nielsen and Chuang, 2000].

In total, we get an upper bound of

$$R \leq \left\lceil \frac{\pi}{4} \sqrt{\frac{N}{M}} \right\rceil, \quad (2.45)$$

which is of the order  $\mathcal{O}(\sqrt{N/M})$  and therefore gives a quadratic speedup compared to the classical search.

### 2.1.3 Juelich Massively Parallel Ideal Quantum Computer Simulator (JUMPIQCS)

I think there is a world  
market for maybe five  
computers.

*(Thomas Watson, chairman  
of IBM, 1943)*

The *Juelich Massively Parallel Ideal Quantum Computer Simulator (JUMPIQCS)* is a software package for simulating a quantum computer using the quantum circuit model (section 2.1.1). It has been created during the work on this thesis. First ideas for a quantum computer simulator are taken from [De Raedt et al., 2007]. The program code has been written from scratch using the C programming language, following the goal to reach a high performance on various computing architectures, especially High Performance Computing (HPC) systems. Therefore, the code uses optimization techniques, e.g. special memory access schemes, that are described in this section.

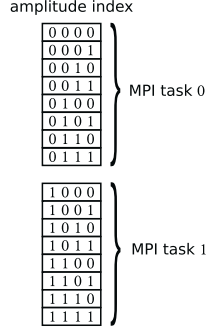
JUMPIQCS is:

- universal: All basic operations for the simulation of any arbitrary quantum algorithm are available.
- efficient: The program code is highly optimized for low memory consumption and high computing performance. It works massively parallel on suitable hardware and shows a very good scaling behavior.
- portable: Written in C it runs on various kinds of hardware ranging from single PCs to high-end parallel supercomputers with distributed and/or shared memory and is also compatible to several software environments (Linux, AIX,...). Standard Message Passing Interface (MPI) [Gropp et al., 1994] is used for communication and optionally OpenMP [Chapman et al., 2007] can be used for parallel processing within each MPI process using multiple threads.

For demonstration purposes a set of algorithms, such as bit adders or the quantum Fourier transform algorithm, are also included in the software package.

JUMPIQCS has been run and tested on different computer architectures, such as the IBM Regatta P690+, the IBM BlueGene/L, and the IBM BlueGene/P. It has been run on the JUMP supercomputer (Jülich Multi Processor) with a maximum system size of 37 simulated qubits using 1024 processors and 3 TB of memory. On the JUGENE system (Jülich BlueGene), the simulation included up to 40 qubits using 49152 processors and 24 TB of memory. To our knowledge, this is currently the largest system size that has been simulated without using approximate methods.

During the course of this work, JUMPIQCS has been gradually extended to simulate non-ideal, error-prone systems (section 2.2) as well as to use quantum error correction techniques (section 2.3).



**Figure 2.10** – Example state vector of a system with  $n=4$  qubits, distributed over  $L=2$  MPI tasks, each containing  $2^M = 2^3 = 8$  complex-valued amplitudes.

**Technical implementation** The simulation of a quantum computer can be done with several approaches. In this work, we deal with pure states, so that the state vector contains the complete information about the quantum state. Since the state vector grows exponentially with the number of qubits, the memory requirements pose a hard limit on possible simulations.

A system with  $n$  qubits has a state vector with  $N = 2^n$  complex-valued amplitudes. Using double precision numbers of 8 bytes length for each of the real and the imaginary part, the memory requirement for storing the state vector is  $2^{n+4}$  bytes.

On computer systems with distributed memory, the state vector is distributed over different memory locations and each processor has access to the local part of the memory only. Message passing is required to gain access to specific parts of the state vector. In this case the  $2^n$  complex-valued amplitudes of the state vector are distributed to local memory parts, each containing  $2^M$  amplitudes, so that we deal with  $L = 2^n/2^M$  MPI processes (see figure 2.10).

**Single qubit operations** As described in section 2.1.1, a single qubit operation on a system of  $n$  qubits acts on one qubit only, while leaving the other qubits untouched. A quantum operation on qubit  $K$  does the transformation

$$U_K = \mathbb{1} \otimes \dots \otimes \mathbb{1} \otimes U_K \otimes \mathbb{1} \otimes \dots \otimes \mathbb{1}. \quad (2.46)$$

Fortunately, it is not necessary to construct the  $2^n \times 2^n$ -dimensional matrix  $U_K$ ; the update of the state vector elements can be done sequentially or in parallel on multi-processor systems. The unitary operation  $U_K$  is decomposed into pairs of two-level unitary gates. It is important to understand that the single qubit operation will usually act non-trivially on all  $2^n$  state vector components.



Let us start with the single processor case, where the whole state vector fits into local memory. The state vector  $|\psi\rangle$  is given by<sup>2</sup>

$$|\psi\rangle = \alpha_0 |0\rangle + \dots + \alpha_{N-1} |N-1\rangle = \begin{pmatrix} \alpha_0 \\ \vdots \\ \alpha_{N-1} \end{pmatrix}, \quad (2.47)$$

with the complex-valued amplitudes  $\alpha_i$  and the computational basis states  $|0\rangle, \dots, |N-1\rangle$ . Using binary notation this can be written as

$$|\psi\rangle = \begin{pmatrix} \alpha_{00\dots 00} \\ \alpha_{00\dots 01} \\ \vdots \\ \alpha_{11\dots 10} \\ \alpha_{11\dots 11} \end{pmatrix}. \quad (2.48)$$

We introduce the notation  $\alpha_{* \dots j_K * \dots *}$ , where the asterisks stand for bits that do not change while  $j_K \in \{0, 1\}$ .

An arbitrary unitary operation on qubit  $K$ ,

$$U_K = \begin{pmatrix} u_{11} & u_{12} \\ u_{21} & u_{22} \end{pmatrix}, \quad (2.49)$$

processes all  $2^n$  elements of the state vector, i.e.

$$U_K |\psi\rangle = |\psi'\rangle = \begin{pmatrix} \alpha'_{00\dots 00} \\ \alpha'_{00\dots 01} \\ \vdots \\ \alpha'_{11\dots 10} \\ \alpha'_{11\dots 11} \end{pmatrix}, \quad (2.50)$$

where

$$\alpha'_{* \dots 0_K * \dots *} = u_{11} \cdot \alpha_{* \dots 0_K * \dots *} + u_{12} \cdot \alpha_{* \dots 1_K * \dots *} \quad (2.51)$$

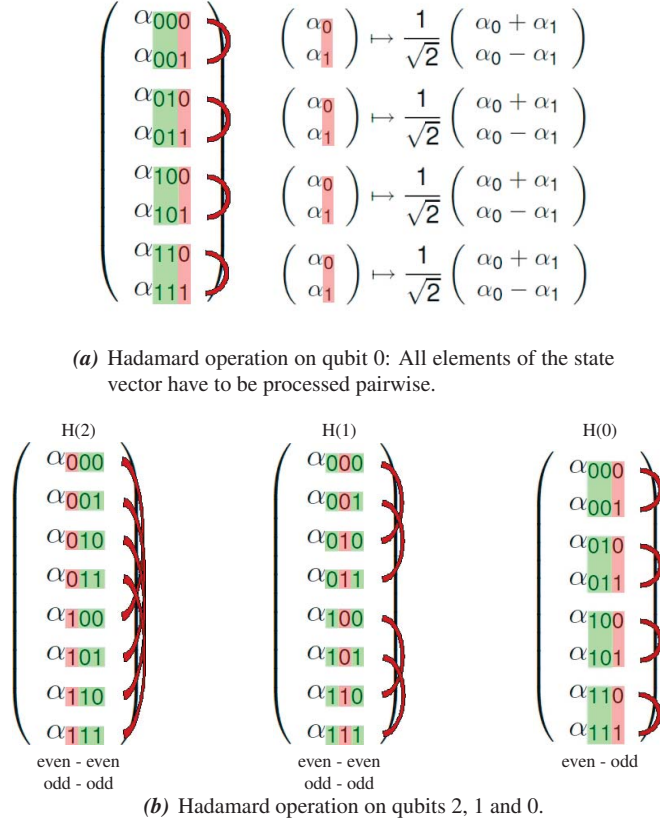
and

$$\alpha'_{* \dots 1_K * \dots *} = u_{21} \cdot \alpha_{* \dots 0_K * \dots *} + u_{22} \cdot \alpha_{* \dots 1_K * \dots *}. \quad (2.52)$$

If  $U_K$  is diagonal, each  $\alpha'_{* \dots 0_K * \dots *}$  is only dependent on  $\alpha_{* \dots 0_K * \dots *}$  and each  $\alpha'_{* \dots 1_K * \dots *}$  only on  $\alpha_{* \dots 1_K * \dots *}$ . In the case of the phase gate  $S = \begin{pmatrix} 1 & 0 \\ 0 & i \end{pmatrix}$  for example,  $u_{11} = 1$ , and only half of the state vector elements have to be processed.

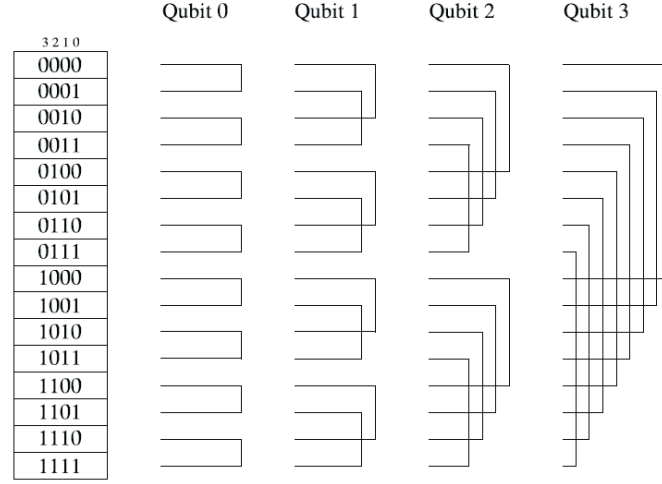
The state vector is stored sequentially in memory. The distance or stride between element  $\alpha_{* \dots 0_K * \dots *}$  and  $\alpha_{* \dots 1_K * \dots *}$  is  $2^K$ . Figure 2.11 shows an example with  $n = 3$  qubits and  $U_K = H_K = 2^{-1/2} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}$ .

<sup>2</sup>In the following, we will work with the standard basis. We will identify the abstract representation of a vector with its coordinate representation and a linear map with its transformation matrix using the chosen basis.

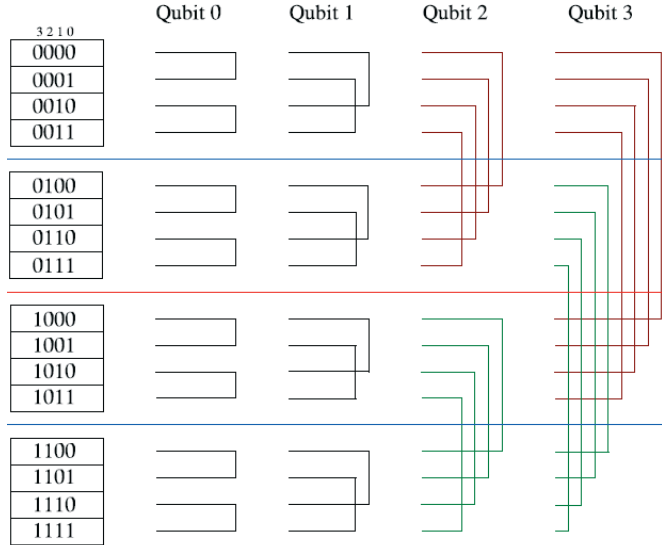


**Figure 2.11** – Memory access scheme for the example of a Hadamard operation on a three qubit system. The state vector is stored and indexed sequentially in memory. The operation can be decomposed into pairwise two-level operations (a). The memory access stride between two pairs of state vector elements depends on the target qubit of the operation (b). The operation on qubit 0 combines successive elements of the state vector, whereas operations on higher qubits link even numbered elements with even numbered ones and odd numbered state vector elements with odd numbered ones only.

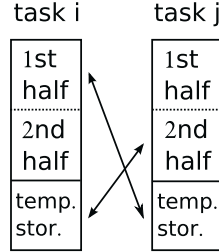
Figure 2.12 shows the different pairs of state vector elements that have to be processed for a general single qubit operation on a system of  $n = 4$  qubits. In the case of a distributed memory system, where the state vector is distributed over  $L > 1$  processes, two different cases have to be taken into consideration. The first one is  $2^K \leq 2^M$  or  $K \leq M$ , so that elements which have to be combined all lie on the same local memory. If  $K > M$ , two elements from distant local memories have to be combined and communication is necessary. Figure 2.13 shows a small example with  $n = 4$  qubits.



**Figure 2.12** – State vector example for  $n=4$  qubits. Pairs of state vector elements that are combined are connected by lines. The pattern changes with the target qubit of the operation.



**Figure 2.13** – State vector example for  $n=4$  qubits on a distributed memory system with  $L = 4$  processes. Pairs of state vector elements that are combined can lie on the same local memory (operation on qubit 0 or 1) or on remote memory locations (operation on qubit 2 or 3), so that communication becomes necessary.



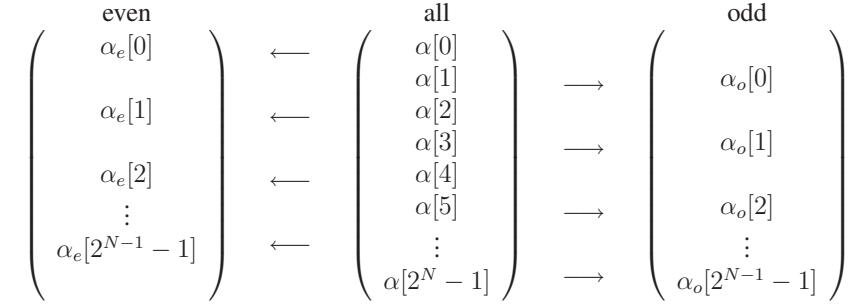
**Figure 2.14** – Communication scheme for the exchange of distributed memory parts. Each MPI task sends half of its local memory to the proper communication partner. After computation involving only local memory, the updated information is sent back.

If  $K > M$ , we use the following communication scheme: Communication is always pairwise, but instead of sending single state vector elements to each other, which substantially increases the communication overhead, blocks of data are sent. Each MPI task sends half of its local memory to the proper communication partner. The two-level operations can then be done locally and the result is communicated back. By using this scheme a native load balancing is achieved, since the computational effort is evenly distributed to the different processes. In general, each single qubit operation involves  $2L$  pairwise exchanges of data blocks, each with  $2^{M-1}$  entries (see figure 2.14). Besides of  $2^{n+4}$  bytes for storing the state vector information, we need an additional temporary storage of  $2^{n+3}$  bytes for doing operations, so that our simulator needs  $3 \cdot 2^{n+3}$  bytes altogether for its operation.

The corresponding communication partners can be determined from the stride  $2^K$  between elements of the state vector, the size  $2^n$  of the state vector and the number of tasks  $L = 2^n / 2^M$ . The distance between corresponding tasks is then given by  $2^{K-M}$ . The core algorithm for looping over every task is given in listing D.1 (see appendix D). This routine ensures that each task finds its correct corresponding communication partner, accounting for possible interleaving patterns of communication processes (see figure 2.12).

**Even/odd-splitting speedup** Note that a single qubit operation on qubit 0 connects successive elements of the state vector, whereas operations on higher qubits link even numbered elements with even numbered ones and odd ones with other odd numbered elements of the state vector only (see figure 2.11). The performance of the code can be improved by splitting up the state vector into two consecutive parts, one containing only the even numbered elements of the state vector and the other containing the odd numbered ones (see figure 2.15). This immediately minimizes the memory access stride by a factor of two for all operations except the ones on qubit 0 and therefore resulting in an average speedup of about 30% in run-time.<sup>3</sup>

<sup>3</sup>Measurement of single qubit operations on specific hardware, in this case the Jülich Multi Processor (JUMP) system.



**Figure 2.15** – Splitting of the state vector in even and odd parts reduces the access strides in memory and saves about 30% of run-time on average.

**Two qubit operations: the controlled-NOT gate** The most important two qubit gate is the controlled-NOT (CNOT) gate, which in conjunction with single qubit operations is sufficient for universal quantum computation (see section 2.1.1). The implementation of the CNOT gate in the JUMPIQCS library is similar to that of single qubit gates. We denote a CNOT operation with control qubit  $C$  and target qubit  $T$  with  $\text{CNOT}(C, T)$ . Then

$$\text{CNOT}(C, T) |\psi\rangle = |\psi'\rangle = \begin{pmatrix} \alpha'_{00\dots00} \\ \alpha'_{00\dots01} \\ \vdots \\ \alpha'_{11\dots10} \\ \alpha'_{11\dots11} \end{pmatrix}, \quad (2.53)$$

where the updated amplitudes  $\alpha'$  are calculated according to

$$\alpha'_{*...*0_C*...*0_T*...*} = \alpha_{*...*0_C*...*0_T*...*}, \quad (2.54)$$

$$\alpha'_{*...*0_C*...*1_T*...*} = \alpha_{*...*0_C*...*1_T*...*}, \quad (2.55)$$

$$\alpha'_{*...*1_C*...*0_T*...*} = \alpha_{*...*1_C*...*1_T*...*}, \quad (2.56)$$

$$\alpha'_{*...*1_C*...*1_T*...*} = \alpha_{*...*1_C*...*0_T*...*}. \quad (2.57)$$

Equations (2.54) and (2.55) denote the identity operation, since the state of the control qubit is not changed, whereas equations (2.56) and (2.57) describe the flip of the target qubit.

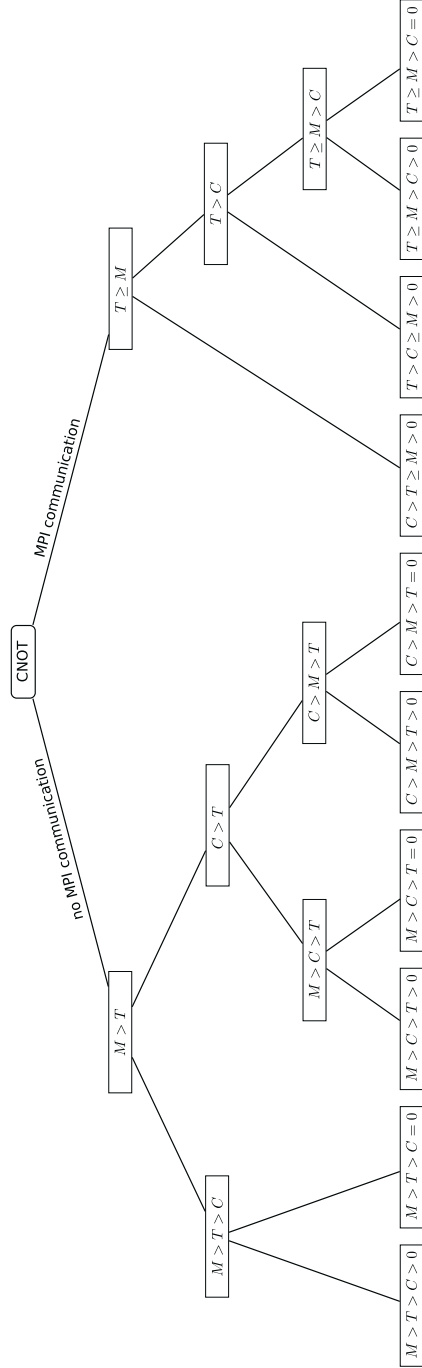
The description above is sufficient for the determination of the proper amplitudes of the state vector which have to be modified. Nevertheless, one can exploit the regular access structure of the state vector amplitudes to optimize memory access. An optimized version of the CNOT gate has been developed as part of this thesis.

There are regular patterns which depend on the following parameters:

- Number of qubits  $n$ .

- Number of MPI tasks  $L$ .
- Control qubit  $C$ .
- Target qubit  $T$ .

By analyzing the occurring patterns, an efficient CNOT implementation can be derived. A CNOT operation leaves half of the  $2^n$  state vector elements untouched, while interchanging the  $2^{n-1}$  amplitudes of the other half. By skipping over the irrelevant state vector entries in advance and working on the others blockwise a significant speedup of the manipulations is achievable. An aggravating circumstance is that these patterns depend on several parameters, so that one has to distinguish several cases. Figure 2.16 gives an overview of all possible cases.



**Figure 2.16** – Possible cases of different memory access schemes. The case depends on the values of the parameters for control qubit  $C$ , target qubit  $T$ , memory size of a single MPI task  $2^M$  and their mutual relations.

A. First of all, the stride between two elements, that have to be interchanged, is always  $2^T$ . If  $M > T$ , i.e., if the size of the local memory blocks are larger than this stride, there is no MPI communication necessary and all operations can be done locally. The core routine for this case is a double nested loop, which can completely describe the occurring access patterns (listing D.2, appendix D). The loop counter variables are denoted  $k$ ,  $l$  and  $m$  and the notation  $j1 = 2^T$  and  $j1 = 2^C$  is used. This ensures an access pattern that works as much as possible on consecutive memory areas to speed up calculation time. The explicit distinction between  $T = 0$  and  $T \neq 0$  is necessary due to our even/odd-splitting scheme.

The optimal algorithm distinguishes between the cases:

1. The control qubit is larger than the target qubit,  $C > T$ . Depending on the size of the local memory, one still has to distinguish between:
  - a.  $T < C < M$ : In this case  $kstart = j1$ ,  $kmax = 2^M - 2^C$  and  $lmax = 2^C - 2 * 2^T$ .
  - b.  $T < M < C$ : In this case the double nested loop reduces to a single nested loop, but there are tasks that are idling and can be skipped. The MPI tasks with ranks  $myrank$ , which fulfill the condition

$$(myrank / (2^{C-M})) \% 2 == 0 \quad (2.58)$$

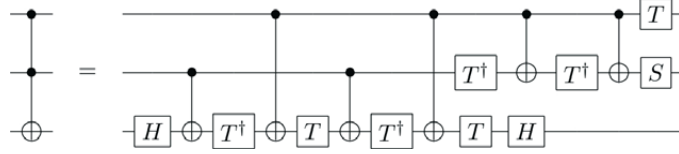
can be leaped over, while the other tasks just execute the  $k$ -loop once with the parameter  $lmax = 2^M - 2 * 2^T$ .

2. The control qubit is smaller than the target qubit,  $C < T$ , while  $T < M$ , so this case simplifies to the fact that every task has to do the same calculations on different parts of the state vector. Again, one has to account for operations with qubit 0 involved, because of the even/odd-splitting scheme. The core loop in this case is found in listing D.3.

B. In the other case, if  $M \leq T$ , MPI communication between different tasks is unavoidable. Again, we have to distinguish two sub-cases:

1.  $M < T < C$ : Here no loops are necessary. The only task is to determine the proper communication partners. The check for  $T = 0$  is not necessary, because that case is covered by case A.1. We have to distinguish three kinds of tasks, those where nothing has to be done and those which send and receive data among each other, labeled first and second communication partner. Those can be determined efficiently with a nested if-case as the code segment in listing D.4 shows (with notation:  $sw = 2^{C-M}$  and  $sw2 = 2^{T-M}$ ).
2.  $C < T$ : This case comprises two subcases. Both can be dealt with by using the double nested loop structure and interleaving it with the algorithm for the determination of the communication partners as shown before in listing D.4.
  - a.  $C < M < T$ : Set  $kstart = j1$ ,  $kmax = j1$ ,  $lmax = 2^M - 2 * j1$ ,  $mmax = j1 - 1$ ;





**Figure 2.17** – Toffoli gate and decomposition into one- and two-qubit gates.

- b.  $M \leq C < T$ : Set  $kmax = 0$ ,  $lmax = 0$ ,  $mmax = 2^M - 1$  and if  $(myrank/sw)\%2 == 0$  set  $kstart = j1$  (k-loop is omitted) and otherwise set  $kstart = 0$  (do a single  $k$ -iteration). The relevant code section is shown in listing D.5.

By using this implementation which exploits the regular structure of the operations on the state vector an average performance increase of 25% could be gained compared to earlier implementations of the CNOT gate [Pomplun, 2005]. The benchmark results are shown in section 2.1.3.

**Three qubit operations: the Toffoli gate** Although single qubit operations and the controlled-NOT gate are sufficient for universal quantum computation, the performance of quantum circuits can be improved by using a 3-qubit gate called Toffoli gate. The Toffoli gate is similar to the CNOT gate, but uses two control bits and one target bit. Sometimes it is denoted as  $C^2NOT$  gate and corresponding generalizations are called  $C^nNOT$  gates. Figure 2.17 shows the Toffoli gate and how it can be composed out of CNOT and single qubit gates. The unitary matrix describing a Toffoli gate operation is given by

$$C^2NOT = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \end{pmatrix}. \quad (2.59)$$

A Toffoli gate flips the target qubit if and only if both control qubits are set to one.

The restriction to single qubit operations and the CNOT gate for universal quantum computation is sufficient, but for some problem cases it is much more convenient to choose another set of basic gates, such as the Toffoli gate and single qubit gates. Taking into account three qubit gates can simplify quantum computations, when complex sequences of two-qubit gates can be replaced by three qubit Toffoli gates. Therefore, the Toffoli gate has also been implemented into the JUMPIQCS package.

The Toffoli gate with control qubits  $C_1$  and  $C_2$  and target qubit  $T$  works on an arbitrary state vector according to

$$C^2NOT(C_1, C_2, T) |\psi\rangle = |\psi'\rangle = \begin{pmatrix} \alpha'_{00\dots00} \\ \alpha'_{00\dots01} \\ \vdots \\ \alpha'_{11\dots10} \\ \alpha'_{11\dots11} \end{pmatrix}, \quad (2.60)$$

where the new amplitudes are given by

$$\alpha'_{*...*0_{C_1}*...*0_{C_2}*...*0_T*...*} = \alpha_{*...*0_{C_1}*...*0_{C_2}*...*0_T*...*}, \quad (2.61)$$

$$\alpha'_{*...*0_{C_1}*...*0_{C_2}*...*1_T*...*} = \alpha_{*...*0_{C_1}*...*0_{C_2}*...*1_T*...*}, \quad (2.62)$$

$$\alpha'_{*...*0_{C_1}*...*1_{C_2}*...*0_T*...*} = \alpha_{*...*0_{C_1}*...*1_{C_2}*...*0_T*...*}, \quad (2.63)$$

$$\alpha'_{*...*0_{C_1}*...*1_{C_2}*...*1_T*...*} = \alpha_{*...*0_{C_1}*...*1_{C_2}*...*1_T*...*}, \quad (2.64)$$

$$\alpha'_{*...*1_{C_1}*...*0_{C_2}*...*0_T*...*} = \alpha_{*...*1_{C_1}*...*0_{C_2}*...*0_T*...*}, \quad (2.65)$$

$$\alpha'_{*...*1_{C_1}*...*0_{C_2}*...*1_T*...*} = \alpha_{*...*1_{C_1}*...*0_{C_2}*...*1_T*...*}, \quad (2.66)$$

$$\alpha'_{*...*1_{C_1}*...*1_{C_2}*...*0_T*...*} = \alpha_{*...*1_{C_1}*...*1_{C_2}*...*0_T*...*}, \quad (2.67)$$

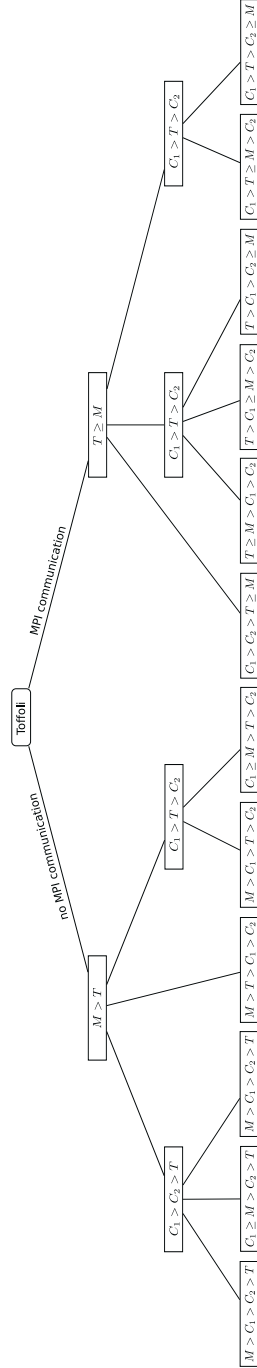
$$\alpha'_{*...*1_{C_1}*...*1_{C_2}*...*1_T*...*} = \alpha_{*...*1_{C_1}*...*1_{C_2}*...*1_T*...*}, \quad (2.68)$$

again using the asterisks to indicate bits of fixed value.

The implementation of the Toffoli gate in the JUMPIQCS library is similar to the implementation of the CNOT operation. The Toffoli gate performs non-trivial operations only on a fourth of the state vector amplitudes. Determining the position of these amplitudes, i.e. the MPI tasks and positions within these tasks, is more complicated than in the CNOT case, because there are more variants of memory access pattern, due to the additional control qubit. The algorithm depends on the control qubits  $C_1$  and  $C_2$ , the target qubit  $T$ ,  $M$ , which determines the size of the MPI tasks, and their relations to each other. Assuming, without loss of generality, that the control qubits  $C_1$  and  $C_2$  fulfill the condition  $C_1 > C_2$ , one has to distinguish between two main cases:

1.  $M > T$ , where no communication is needed and
2.  $M \leq T$ , where communication is necessary,

and subcases thereof. The tree of possibilities shown in figure 2.18 gives a summary of the possible cases, each leading to a different memory access scheme. For the sake of clarity the distinction between the lowest value equal to zero or non-zero is being omitted. In order to realize even/odd-splitting these cases must be distinguished in the code.



**Figure 2.18** – Possible cases of different memory access schemes for the Toffoli gate depending on the values of the parameters for control qubits  $C_1$  and  $C_2$ , target qubit  $T$  and memory size of a single MPI task  $2^M$  and the relations between them.

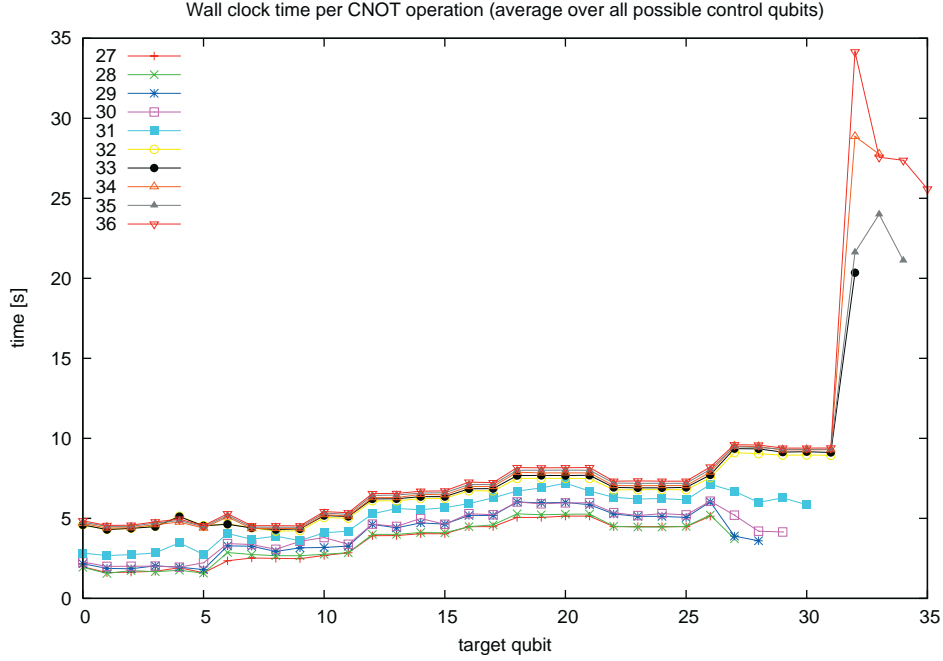
**Benchmarking the CNOT operation** The run-time of the CNOT operation depends heavily on the chosen parameters. Therefore, an exact description of the chosen parameters is required. In general, operations involving lower qubits are faster than operations involving higher qubits, because of different memory access strides. In our simulator the differences in runtime are intrinsic and cannot be avoided.

Benchmarks have been run on the JUMP system (Jülich Multi Processor) of the Jülich Supercomputing Centre. The JUMP system<sup>4</sup> is a cluster computer consisting of 41 IBM p690+ racks each containing 32 IBM Power4+ CPUs running at 1.7 GHz. Its peak performance is 8.9 Teraflop/s and it has an aggregated main memory of 5.2 TB. There are two Power4+ processors on one chip and four chips form a Multi Chip Module (MCM). Four MCMs form a SMP<sup>5</sup> node, where each processor has access to the shared memory of 128 GB. The 41 racks are connected by IBM's High Performance Federation Switch [Johnston and King-Smith, 2005].

Figure 2.19 shows a benchmark where the wall clock time has been measured depending on the target qubit for several system sizes. This is a weak scaling benchmark, where the local system size has been fixed to the maximal possible system size. In case of the JUMP system this is equivalent to 27 qubits on one processor. The state vector of a 27 qubit system has  $2^{27}$  complex-valued amplitudes, which are saved as double precision real numbers for each the real and the imaginary part. This means that the storage of a 27 qubit system state vector requires  $2^{27+3+1}$  Bytes, i.e. 2 GB of memory. An additional 1 GB of memory is needed for doing operations on the state vector. In total, a single processor of the JUMP system can operate on a system of 27 qubits and each additional qubit doubles the resources that are required. Since the control qubit does also influence the memory access patterns and therefore the measured timings, the measurement result for each target qubit is actually an average over multiple measurements with all possible combinations of the control qubit while keeping the target qubit fixed. The plot shows results for various system sizes. The increase in runtime depending on the system size is due to the non-perfect scaling of the JUMPIQCS CNOT code on the JUMP system. The slight increase of the wall clock time depending on the target qubit can be related to the increasing memory access stride. Inter-processor communication (necessary for target qubits 27 to 31) does not slow down the performance for system sizes up to 31 qubits. An increase of the wall clock time can be seen for system sizes larger than 31 qubits. The huge increase going from target qubit 31 to target qubit 32 is a result of the non-uniform memory access architecture of the JUMP system. Up to target qubit 31 only intra-node communication is necessary, while going to target qubit 32 requires inter-node communication, i.e. communication spanning over two physical racks using the IBM HPS Cluster interconnect [Johnston and King-Smith, 2005].

<sup>4</sup>During the work on this thesis, the system has been replaced by another intermediate supercomputer, while retaining the name JUMP. The new system is a IBM Power6 575 system with 14 SMP nodes, each with 32 SMT processors. The peak performance is 8.4 Teraflop/s and it has an aggregated memory of 1.8 TB.

<sup>5</sup>Symmetric multiprocessing



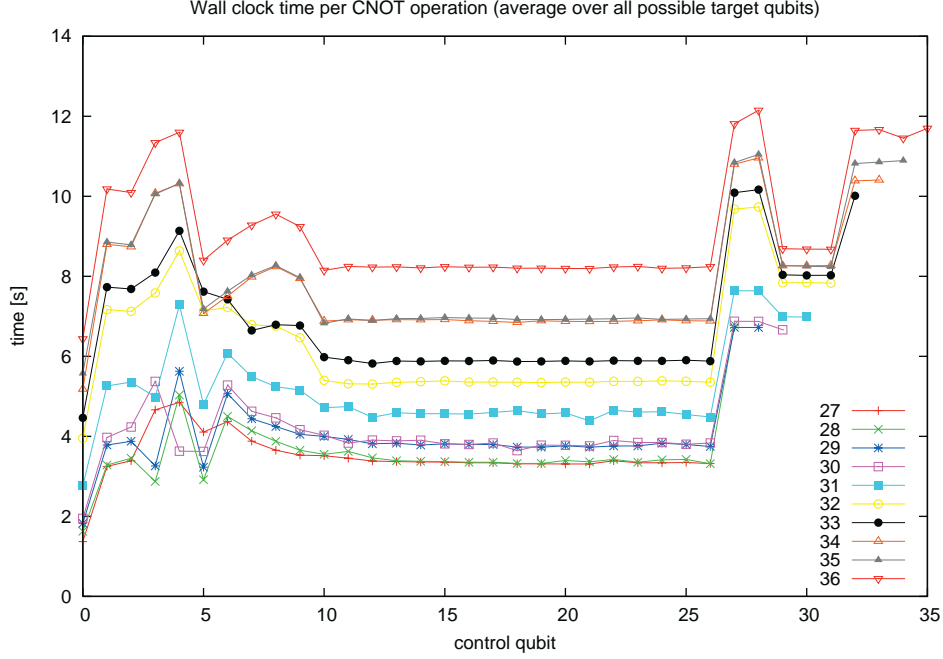
**Figure 2.19** – Benchmark results for the controlled-NOT operation on the JUMP IBM p690+. The dependency of the wall clock time on the target qubit is shown, where each measurement point is the average over multiple measurements with all possible parameters for the control qubit. The plot shows the results for various system sizes. Lines are guides to the eye.

The measured<sup>6</sup> minimal latency between two MPI tasks residing on the same physical node is  $2.1 \mu\text{s}$ , whereas the measured inter-node latency is  $5.8 \mu\text{s}$ . The measured maximum intra-node uni-directional MPI bandwidth is  $3.1 \text{ GB/s}$ , while the inter-node MPI bandwidth is slower at  $2.7 \text{ GB/s}$  only. Minor fluctuations are most probably related to cache effects.

Figure 2.20 shows a similar plot, but this time depending on the control qubit and averaging the timings over all possible target qubits. The dependency on the control qubit is evident and again, the jumps at qubit 27 and qubit 32 are due to the non-uniform memory architecture of the JUMP system. The differences in run-time can be related to differences in memory access patterns in combination with cache effects.

Since the timings depend significantly on the parameters chosen for control and target qubit an averaging over all possible combinations of control and target qubits has been performed and the results are shown in figure 2.21. Since this is a weak scaling benchmark, the ideal case would be a constant, while in reality the times increase slightly with every doubling

<sup>6</sup>Measurements were done using the Pallas MPI Benchmarks Suite V2.2 (now called Intel®MPI Benchmarks)

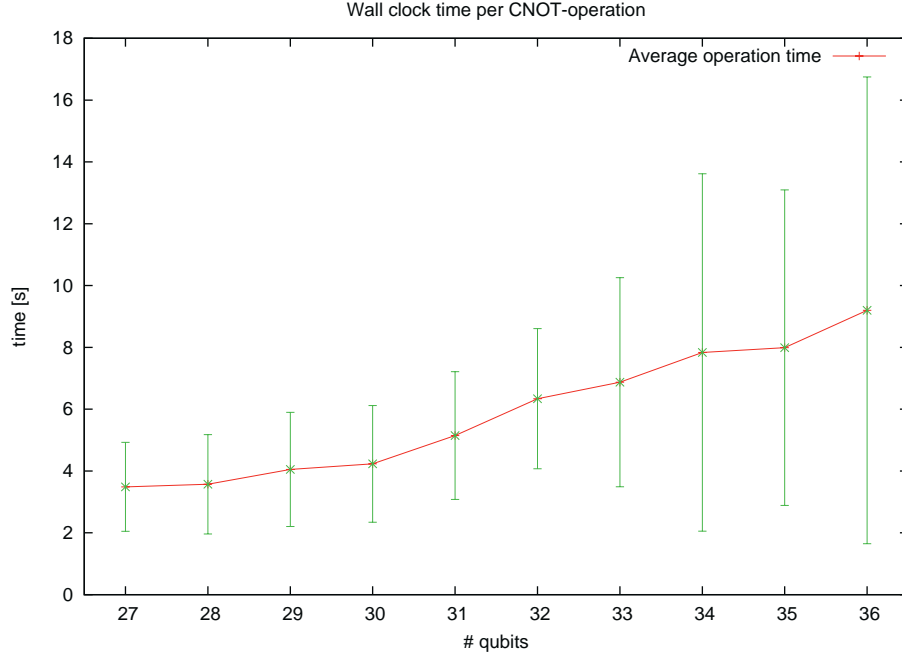


**Figure 2.20** – Benchmark results for the controlled-NOT operation on the JUMP IBM p690+. The dependency of the wall clock time on the control qubit is shown, where each measurement point is the average over multiple measurements with all possible parameters for the target qubit. The plot shows the results for various system sizes. Lines are guides to the eye.

of processors and memory. To show that this increase is mainly due to the inter-node communication, the same analysis as in figure 2.21 has been done again, but this time the target qubits are broken down into intervals (see figure 2.22). This plot shows that the increase in runtime is mainly due to the operations on target qubits larger than 31 leading to a memory access scheme with a large stride involving inter-node communication.

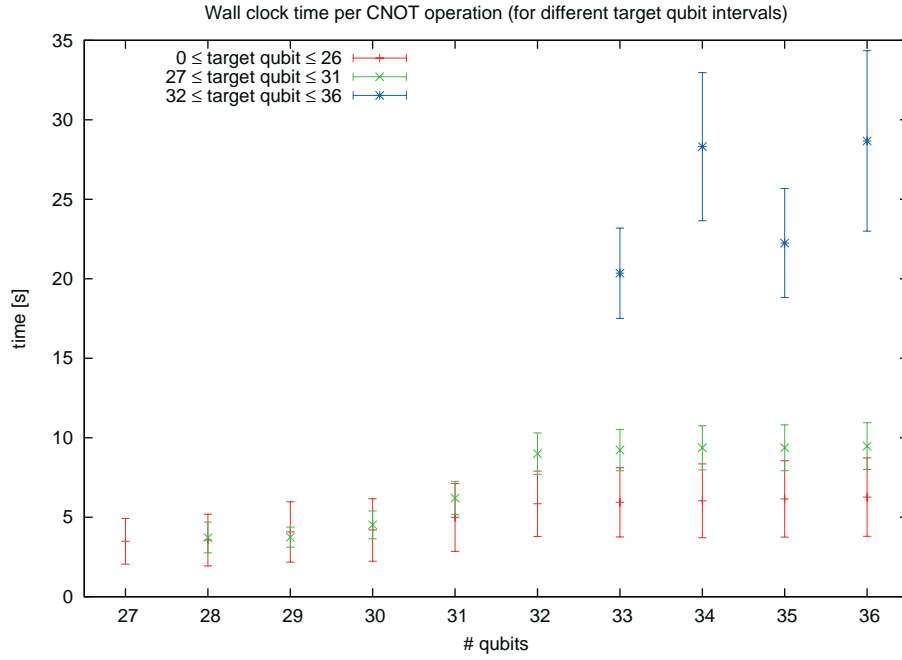
In order to see the actual dependency of the run-time on the individual parameters of control and target bit, one example has been included, where no averaging over control or target qubit has been done (figure 2.23). It becomes evident that the dependency on the target qubit, which defines the memory access stride, has the strongest impact, especially when it comes to inter-node communication.

The benchmark results show that the controlled-NOT operation does not exhibit an ideal scaling behavior. The performance depends considerably on the parameters for control and target qubit. Usually operations on lower target qubit values are faster, so in the following, we construct our quantum algorithms in such a way, that we favor operations on lower-valued qubits. Especially on non-uniform memory access architectures the drops in



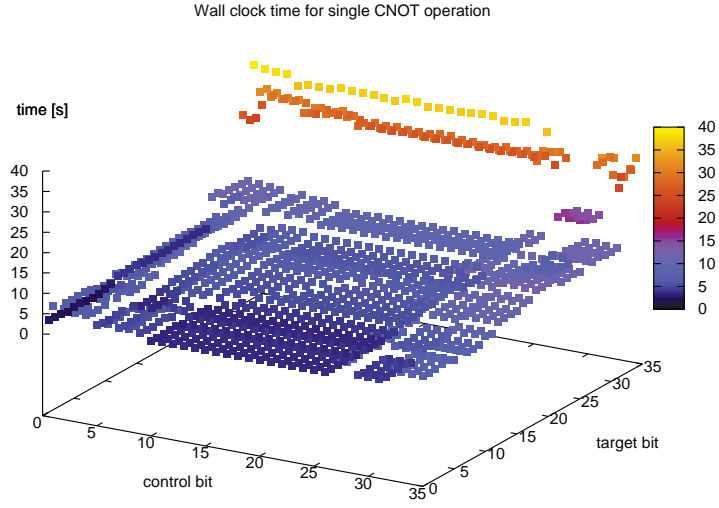
**Figure 2.21** – Benchmark results for the controlled-NOT operation on the JUMP IBM p690+. The dependency on the system size is shown, where each measurement point is the average over multiple measurements with all possible parameters for the control qubit and the target qubit. The error bars indicate the standard deviation of the occurring execution times (for different values of the control qubit and the target qubit). The line is a guide to the eye.

performance can be considerable at certain limits. This has to be taken into consideration when optimizing quantum algorithms for high performance.



**Figure 2.22** – Benchmark results for the controlled-NOT operation on the JUMP IBM p690+. The dependency on the system size is shown, while it has been averaged over the measurement times of all possible combinations for the control qubit and the target qubit. The analysis has been done for different intervals of the target qubit. The error bars indicate the standard deviation of the occurring execution times. The increase in runtime can be mainly related to operations on target qubits larger than 31, because this results in memory access patterns with large stride involving inter-node communication.





**Figure 2.23** – Benchmark results for the controlled-NOT operation on the JUMP IBM p690+. Dependency of the wall clock time on the parameters for the control qubit and the target qubit. The operation time is mainly defined by the value of the target qubit, which specifies the memory access pattern. Especially for a non-uniform memory access architecture like the JUMP system, certain operations are considerably slower when involving communication between processors on different nodes.

## 2.2 Error-Prone Quantum Computer Simulations

The JUMPIQCS library is a complete simulator for an ideal quantum computer. In this case the internal method of operation for each component does not impact the logical operation of the whole quantum computer. However, for a realistic quantum computation device it is important to consider the physics of the basic building blocks of that device. In the idealized case, each operation is done instantaneously and with no error at all. An operation on a single qubit does not affect other qubits of the overall system. In the real world there are deviations from perfect spin rotations as well as unwanted interactions through spin-spin couplings and couplings to the environment. The loss of quantum coherence, i.e. the transition from quantum states to classical states, is called decoherence. It occurs whenever a system interacts with additional external degrees of freedom, and this is the main obstacle in the construction of large-scale quantum computers. Therefore, realistic simulations that involve decoherence are essential for the verification and optimization of real quantum computation devices.

One possibility is to use a device-specific Hamiltonian for the simulation of a particular quantum computation device. Such an approach is used in chapter 3 for analyzing the dynamics of an ion trap quantum computation device. However, in this chapter, we take another approach, i.e., we extend the gate level simulator JUMPIQCS with an error model (section 2.2.1), that describes possible errors at the level of single quantum gates.

Many problems are too complex to be solved with theoretical analytical methods (without making unrealistic simplifications and concentrating on very special cases). Computer simulations can cope with this problem. Here, we are using the JUMPIQCS code with error model extension to analyze the behavior of a noisy quantum computer by studying the robustness of several algorithms prone to noise.

### 2.2.1 Error Model

As an extension to the *Juelich Massively Parallel Ideal Quantum Computer Simulator (JUMPIQCS)* we implement an error model for operational imperfections and decoherence. The error model is integrated into the JUMPIQCS. It stays within the gate level simulation framework.

**Decoherence** Decoherence is a problem that will affect every quantum computation device. The superposition states that every quantum computer deals with are fragile and any interaction with the environment will lead to a decay of quantum information and therefore to errors in the computation. Since no device can be perfectly isolated from the environment decoherence as a source of error is in principle inevitable.

We implemented the so-called depolarizing channel [Nielsen and Chuang, 2000] as a suitable error model for the simulation of decoherence. In this model general errors are described as linear combinations of basic errors, given by the Pauli matrices  $\sigma_x$ ,  $\sigma_y$  and  $\sigma_z$ . A  $\sigma_x$ -error,  $\sigma_x = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}$ , can be identified as a bit flip, a  $\sigma_z$ -error,  $\sigma_z = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}$ , as a phase flip and  $-i\sigma_y = \sigma_x\sigma_z$  can be characterized as a combined bit and phase flip.

The depolarizing channel assumes an equal probability of  $p/3$  for the occurrence of each error and an error-free evolution with probability  $1 - p$ . The occurring errors are locally and sequentially independent.

We use this error model to account for decoherence affecting our system, so we introduce a timescale defined by the single qubit operation time  $\delta t$  and let the decoherence operation affect each qubit after each timestep  $\delta t$ .

Since decoherence results in mixed states, but our simulator deals with pure states only, we have to sample many runs using statistically independent random number sequences. The random numbers are drawn from a uniform distribution using the GNU Scientific Library with the Mersenne Twister MT19937 algorithm [Galassi et al., 2006]. The result is an ensemble of pure states weighted by their corresponding probabilities of occurrence. The use of stochastically independent error locations is justified by the observation that the behavior of a quantum network can be represented as a mixed sum of networks with linear error operators placed at each error location. This approach is also known as error expansion [Knill et al., 1998b].

**Operational imprecisions** Every experiment in the lab will suffer from operational imprecisions, as quantum superpositions deal with continuous variables and one cannot control quantum operations with infinite accuracy and precision. We implemented a model for operational imprecisions that includes unitary over-rotations<sup>7</sup> to resemble the errors occurring in the lab. It is not specific to a certain realization of a quantum computation device, but its parameters can be adapted to a range of experimental setups.

The Pauli matrices  $\sigma_x$ ,  $\sigma_y$  and  $\sigma_z$  are the generators of rotations on the Bloch sphere and each single qubit operation can be written as an element of the complex rotational group  $SU(2)$ ,

$$R_{\vec{n}}(\theta) = e^{-i\theta\vec{n}\cdot\vec{\sigma}/2} = \cos\left(\frac{\theta}{2}\right) \mathbb{1} - i \sin\left(\frac{\theta}{2}\right) (n_x\sigma_x + n_y\sigma_y + n_z\sigma_z), \quad (2.69)$$

with  $\vec{n}$  being a three dimensional unit vector,  $\vec{\sigma}$  the three component vector of Pauli matrices, and  $\theta$  denoting the rotation angle.

There are several choices for decomposing a general rotation into a product of rotations about the axes. For example one could decompose a general rotation into a product of

---

<sup>7</sup>We use the term over-rotation in the sense of imperfect rotation for both over- and under-rotations.

rotations about the three axes,

$$R_{\vec{n}}(\theta) = R_x(\theta_x)R_y(\theta_y)R_z(\theta_z), \quad (2.70)$$

or one could choose the Euler angle convention with

$$R_{\vec{n}}(\theta) = R_z(\theta'_z)R_y(\theta_y)R_z(\theta_z). \quad (2.71)$$

For our simulations we chose a model similar to that in [Niwa, 2002] where we only consider plane rotations around the  $y$ -axis and the  $z$ -axis, i.e., each single qubit quantum operation can be constructed from plane rotations in the  $x$ - $z$ -plane,

$$R(\theta) = \begin{pmatrix} \cos(\theta) & -\sin(\theta) \\ \sin(\theta) & \cos(\theta) \end{pmatrix}, \quad (2.72)$$

and phase shifts

$$P(\phi) = \begin{pmatrix} 1 & 0 \\ 0 & e^{i\phi} \end{pmatrix}. \quad (2.73)$$

This is a restriction of generality, but it is still sufficient to generate every possible rotation on the Bloch sphere. The Hadamard operation for example can be decomposed into

$$H = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix} = R\left(\frac{\pi}{4}\right)P(\pi). \quad (2.74)$$

This decomposition allows to introduce errors  $\epsilon$  in the rotation angles:

$$\theta' = \theta + \epsilon_\theta, \quad (2.75)$$

$$\phi' = \phi + \epsilon_\phi. \quad (2.76)$$

These angle and phase errors  $\epsilon$  are generated from a Gaussian distribution  $\rho$  with mean  $\mu$  and standard deviation  $\sigma$ :

$$\rho(\epsilon) = \frac{1}{\sigma\sqrt{2\pi}} e^{-(\epsilon-\mu)^2/(2\sigma^2)}. \quad (2.77)$$

The Gaussian distribution is generated by the GNU Scientific Library, which uses the Box-Muller algorithm, with the previously mentioned random number generator as an underlying source of randomness [Galassi et al., 2006]. This gives uncorrelated stochastic errors.

One should be aware that the results of our simulations were generated with equal Gaussian distributions for the errors around both axes, i.e.  $\epsilon_\theta = \epsilon_\phi = \epsilon$ , but since we used the definition for our  $y$ -rotations as in equation (2.72) (to allow for a comparison of our results with those in [Niwa, 2002]), the errors for this rotational direction are chosen twice as large as for the  $z$ -rotations. Actually, the relation between the precision of phase gates and  $y$ -rotations depends on the experimental realization, so there is no justification that both errors have to be necessarily of the same size.

The handling of operational errors for two-qubit gates follows a similar path and in our simulation runs we decided to introduce an error of the same size on the target qubit while leaving the control qubit untouched. For future runs there is also the possibility to introduce errors in both the control and target qubit. Again this choice depends on specific experimental realizations and for example in the case of ion trap qubits, a controlled-NOT operation severely affects the control qubit through coupling to the target qubit via the phonon bus (see section 3.1). The analyses we made in this section are for a rather general abstract model which does not account for a specific experimental realization.

**Interplay of error sources** Our error model covers both decoherence errors and operational errors, the main sources of noise in quantum computation. In appendix B we proof that, for a single qubit, both error approaches are actually equivalent to each other for certain choices of parameters. Nevertheless, the motivation to use both sorts of errors stems from the resemblance to real experiments. It is easier to differentiate between errors that are induced by doing operations on a qubit and errors due to interaction with the environment. The strength of each error source can be quite different from each other, so that differentiating between those errors is sensible. The main difference is that the locations of error occurrences within a quantum circuit will differ from each other: While operational errors affect only those qubits on which operations are done, decoherence affects all qubits at each timestep of the algorithm. The error locations within a quantum circuit can differ substantially depending on the quantum algorithm (see sections 2.2.3 and 2.2.4).

### 2.2.2 $H^{2k}$ -Algorithm

The first verification of our error model was made by looking at a very simple algorithm, i.e. a sequence of Hadamard operations. For this simple algorithm the evolution of the density matrix, respectively the state vector, can be determined analytically even under the influence of a single sort of noise.

The Hadamard operation (equation (2.74)) is an involution, i.e., the repeated application of the Hadamard operation gives the identity for an even number of repetitions:

$$H \circ H = \mathbb{I}. \quad (2.78)$$

In the error-free case the repeated application of  $H^2$  does not change the state vector at all. As described in [Niwa, 2002] we calculate the decrease of the fidelity under the influence of either decoherence errors or rotational errors. The fidelity for two pure states  $|\psi\rangle$  and  $|\phi\rangle$  is defined as

$$F = |\langle\psi|\phi\rangle|^2, \quad (2.79)$$

and describes the overlap between two quantum states, giving a distance measure.<sup>8</sup> Our simulations including the noise model rely on statistical distributions of single gate errors, so due to the statistical description our readout is not a single shot readout, but averaged over many experiments. All calculations were made by starting with a well-defined pure state vector. The behavior to noise is then determined by running the algorithm many times with a sample of statistically independent random error events and evaluating the averaged final state. Thus, the fidelity we are measuring is given by

$$F = \frac{1}{m} \sum_{i=1}^m |\langle\psi_C|\psi_i\rangle|^2, \quad (2.80)$$

where  $m$  is the number of experiments, each starting with a different initial seed for the random number generator,  $|\psi_C\rangle$  is the correct output state vector in the error-free case and  $|\psi_i\rangle$  are the single output state vectors of each statistical run.

The density matrix for the sequence of Hadamard operations under the influence of the depolarizing channel at each step  $l$  evolves according to

$$\rho_{l+1} = (1-p)H\rho_l H^\dagger + \frac{p}{3} (\sigma_x H\rho_l H^\dagger \sigma_x^\dagger + \sigma_y H\rho_l H^\dagger \sigma_y^\dagger + \sigma_z H\rho_l H^\dagger \sigma_z^\dagger). \quad (2.81)$$

In case we start with the input state  $|00 \dots 00\rangle$ , i.e. all qubits set to zero, we can calculate the decrease in fidelity analytically. For the single qubit case the density matrix at the beginning  $\rho_0$  is just

$$\rho_0 = \begin{pmatrix} 1 & 0 \\ 0 & 0 \end{pmatrix}, \quad (2.82)$$

<sup>8</sup>There is also an alternative definition of the fidelity  $F$  where the square root is taken, so that it has no interpretation as a transition probability any more (e.g. in [Nielsen and Chuang, 2000]). We will call this  $\sqrt{F}$  the square root fidelity.

and after  $2k$  Hadamard operations it is

$$\rho_{2k} = \frac{1}{2} \begin{pmatrix} 1 + (1 - \frac{4}{3}p)^{2k} & 0 \\ 0 & 1 - (1 - \frac{4}{3}p)^{2k} \end{pmatrix}. \quad (2.83)$$

For the input vector  $|0\rangle$  the entry  $\rho_{00}$  of the density matrix is equal to the fidelity (equation (2.79)).

The generalization to  $n$  qubits gives

$$\rho_{00}^{(2k)} = \left( \frac{1 + (1 - \frac{4}{3}p)^{2k}}{2} \right)^n \quad (2.84)$$

for the  $\rho_{00}$  element of the density matrix, which is equal to the fidelity in this case, after  $2k$  iterations of the Hadamard operation.

A similar analysis can be carried out for isolated operational errors. For a single qubit the density matrix after application of the error-prone Hadamard gates is

$$\rho_{l+1} = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} H(\epsilon_{\theta}, \epsilon_{\phi}) \rho_l H(\epsilon_{\theta}, \epsilon_{\phi})^{\dagger} p(\epsilon_{\theta}) p(\epsilon_{\phi}) d\epsilon_{\theta} d\epsilon_{\phi}, \quad (2.85)$$

with the central Gauss distribution

$$p(\epsilon) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\epsilon^2/(2\sigma^2)}. \quad (2.86)$$

For the input state vector  $|0\rangle$  this leads to

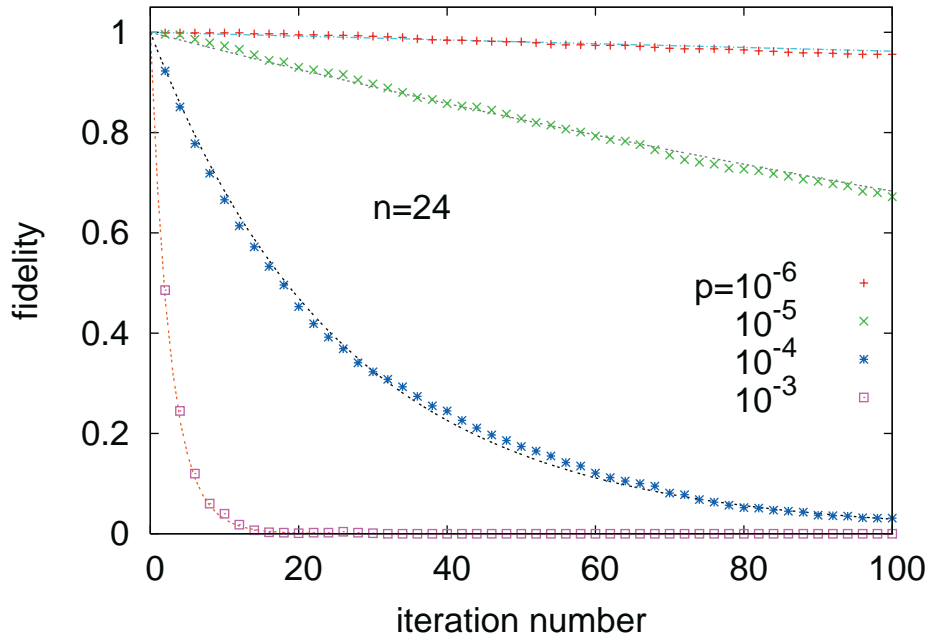
$$\rho_{2k} = \frac{1}{2} \begin{pmatrix} 1 + e^{-\frac{9}{4}\sigma^2 2k} & 0 \\ 0 & 1 - e^{-\frac{9}{4}\sigma^2 2k} \end{pmatrix}. \quad (2.87)$$

Again, the generalization to  $n$  qubits gives the fidelity for repeated application of error-prone Hadamard operations:

$$\rho_{00}^{2k} = \left( \frac{1 + e^{-\frac{9}{4}\sigma^2 2k}}{2} \right)^n. \quad (2.88)$$

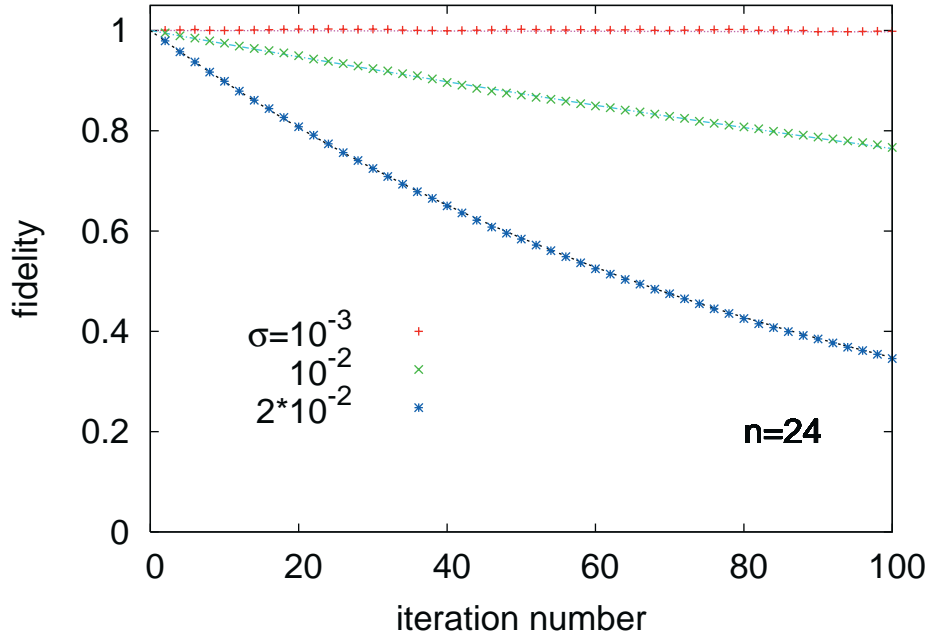
We ran our simulations with this artificial algorithm to assess our simulation package. A simulation result for the depolarizing channel error can be seen in figure 2.24, a result for the unitary over-rotations is shown in figure 2.25. The agreement between the simulation output and the analytical results is very good and verifies our simulation code with included error model.

In the next two sections we use our gate level simulator with verified error model included to analyze two of the most important quantum algorithms: The quantum Fourier transform, which is at the heart of Shor's prime factorization algorithm, and Grover's search algorithm. We examine how these algorithms perform under noisy conditions, investigate their sensitivity and robustness to noise and study how the performance scales with system size.



**Figure 2.24** – Fidelity for a sequence of Hadamard operations depending on the iteration number under the influence of depolarizing channel noise. Results for various error probabilities  $p$  and a system size of  $n = 24$  qubits are shown. Data points are simulation results, while the lines indicate analytical expressions (equation (2.84)). The numerical simulations are well in agreement with the analytical results.





**Figure 2.25** – Fidelity for a sequence of Hadamard operations depending on the iteration number under the influence of unitary operational error noise. The system size is  $n = 24$  qubits. Results for various values of the standard deviation of the Gaussian distributed errors  $\sigma$  are shown. Data points are simulation results, while the lines indicate analytical expressions (equation (2.88)). The numerical simulations are well in agreement with the analytical results.

### 2.2.3 Quantum Fourier Transform

The quantum Fourier transform (QFT) is the kernel of Shor's prime factorization algorithm. A detailed description of the ideal case algorithm can be found in section 2.1.2, and the quantum circuit is shown in figure 2.4.

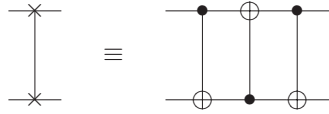
The error-prone Hadamard gates are implemented analogously to equation (2.74) as

$$H_\epsilon = R\left(\frac{\pi}{4} + \epsilon_\theta\right) P(\pi + \epsilon_\phi), \quad (2.89)$$

and the controlled phase shifts are realized as effective single qubit phase shifts with rotation angle  $\phi = 2\pi/2^k$ :

$$P_k = P\left(\frac{2\pi}{2^k} + \epsilon_\phi\right). \quad (2.90)$$

Each swap gate at the end of the algorithm is realized as a sequence of three CNOT gates using the identity shown in figure 2.26.



**Figure 2.26** – Decomposition of the swap gate into three subsequent CNOT gates.

An effective NOT gate (bit flip  $\sigma_x$ ) is decomposed into

$$\sigma_x = R\left(\frac{\pi}{2} + \epsilon_\theta\right) P(\pi + \epsilon_\phi) \quad (2.91)$$

to allow for operational deviations.

We use this model to study the QFT with a set of different parameters and analyze the robustness of the QFT to errors. As a measure of “correctness” of the output state vector we use the error norm  $e^2$ , which is defined as<sup>9</sup>

$$e^2 = \|(|\psi\rangle - |\psi_C\rangle)\|^2 = \langle\psi - \psi_C|\psi - \psi_C\rangle. \quad (2.92)$$

The output state  $|\psi\rangle$  is compared to the correct output state  $|\psi_C\rangle$  that would occur in the error-free case. Again, our measurement result is the average of many individual events, so in fact the error norm of a run with a statistical sampling size  $m$  is

$$e^2 = \frac{1}{m} \sum_{i=1}^m \|(|\psi_i\rangle - |\psi_C\rangle)\|^2 = \frac{1}{m} \sum_{i=1}^m \langle\psi_i - \psi_C|\psi_i - \psi_C\rangle. \quad (2.93)$$

<sup>9</sup>Note that we call  $e^2$  the error norm without taking the square root.

The input vector of the QFT routine is set to  $|00 \dots 00\rangle \hat{=} (10 \dots 00)^T$ , so that the output state vector in the ideal case is the uniform superposition  $|\psi_C\rangle \hat{=} 2^{-n/2}(11 \dots 11)^T$ .

We analyze the system sizes  $n = 8$  and  $n = 16$  qubits using  $m = 10^5$  statistical repetitions and  $n = 24$  using  $m = 10^4$  repetitions in dependence of the decoherence probability parameter  $p$  and the standard deviation  $\sigma$  of the Gaussian distributed operational inaccuracies. The results can be found in figure 2.27. The simulation results show that the quantum Fourier transform is rather robust against operational errors up to a level of  $\sigma \approx 10^{-2}$  ( $\approx 0.6$  degrees). Within the resolution of the statistical fluctuations there is no significant difference between the error-free case and the case with  $\sigma = 10^{-2}$ , even for the largest system that we have examined. In contrast, the error norm increases considerably with increasing system size when  $\sigma > 10^{-2}$ . Figure 2.27 also indicates a sort of plateau behavior, i.e., the error norm gradient is rather flat over a wide range of decoherence probabilities  $p$  and at some point the error norm increases substantially up to the maximum error norm of 2.<sup>10,11</sup> A quantitative analysis of the system size dependency of decoherence errors is shown in figure 2.28. We did also run a simulation for a system of  $n = 32$  qubits with a statistical sample size of  $m = 1000$ . This plot suggests that for the evaluated system sizes a doubling of the system size leads to the necessity of bringing down the error probability about one order of magnitude to maintain a constant error norm level.

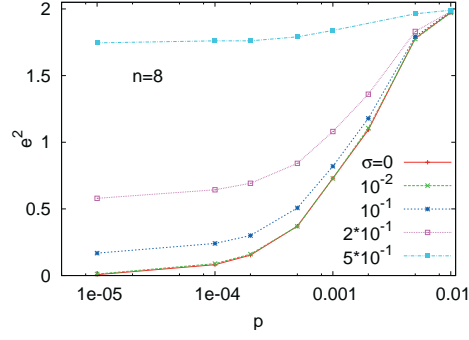
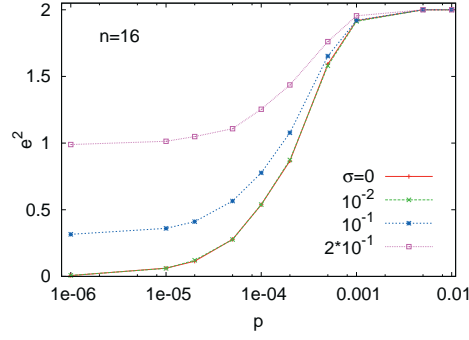
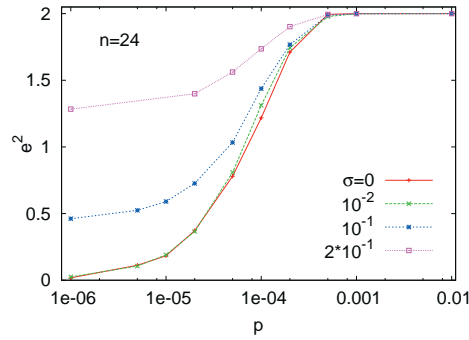
**Visualization of the effects of errors on the QFT algorithm** Using scalar observables such as the error norm or the fidelity is a good way to quantify the impact of noise on the output state vector. To get a more detailed insight into the propagation of errors during the QFT algorithm and to visualize the effect of noise we evaluate the expectation values  $\langle\sigma_x\rangle$ ,  $\langle\sigma_y\rangle$  and  $\langle\sigma_z\rangle$  and plot the resulting vector into the Bloch sphere. This is especially helpful for the prediction of projective measurement results.

Using the notation

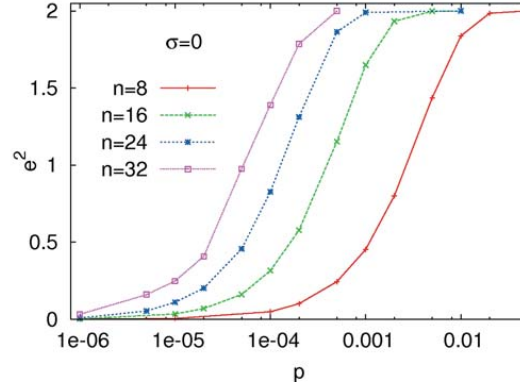
$$|\psi\rangle = \alpha_0 |0\rangle + \alpha_1 |1\rangle \quad (2.94)$$

<sup>10</sup> $e^2 = \|(|\psi\rangle - |\psi_C\rangle)\|^2 = \langle\psi - \psi_C|\psi - \psi_C\rangle = \|\psi\|^2 + \|\psi_C\|^2 - \langle\psi|\psi_C\rangle - \langle\psi_C|\psi\rangle$ , i.e., the error norm is zero if and only if both states are identical, and it is 2 if both state vectors are orthogonal or maximally different. In the noisy limit and for large systems the part of the equally distributed random state vector in the direction of the correct solution is arbitrarily small, so that  $\langle\psi|\psi_C\rangle \rightarrow 0$  and  $e^2 \rightarrow 2$ .

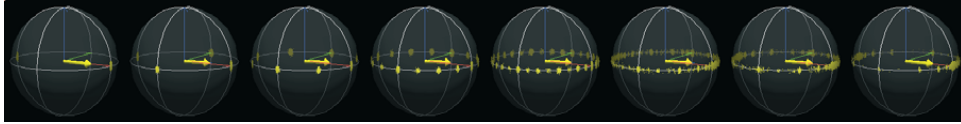
<sup>11</sup>Actually, the error norm does contain information about global phase differences, in contrast to the fidelity, which discards any information about global phases. The error norm  $e^2$  can actually take values up to 4, e.g. for  $|\psi\rangle = -|\psi_C\rangle$ . Global phases do not have a physical meaning and both states must be considered equal. Nevertheless, for the analyses in this section we can use the error norm, because none of the quantum operations in the algorithm will introduce global phases. Later on, for the analyses of quantum error correction codes, we have to take the fidelity as a measure of distance, because we have to omit information about global phases which do occur then.


 (a) system size  $n=8$  qubits

 (b) system size  $n=16$  qubits

 (c) system size  $n=24$  qubits

**Figure 2.27** – Error norm  $e^2$  for the error-prone quantum Fourier transform with varying decoherence probability  $p$  and array parameter  $\sigma$ , the standard deviation of Gaussian distributed operational inaccuracies. Lines are guides to the eye. The curves for  $\sigma = 0$  and  $\sigma = 10^{-2}$  are nearly identical for all system sizes considered.



**Figure 2.28** – Error norm for the quantum Fourier transform depending on the decoherence rate  $p$  for various system sizes. Operational errors are switched off ( $\sigma = 0$ ). Lines are guides to the eye.



**Figure 2.29** – Output state for an eight qubit system after a QFT has been done on the input state  $|00000000\rangle$ . The error model has been driven with the parameters  $\sigma = 10^{-2}$  and  $p = 5 \cdot 10^{-3}$ . Each statistical iteration gives a yellow dot, the ensemble average is represented by the yellow arrow.

the expectation values are given by

$$\langle \sigma_x \rangle = \alpha_0^* \alpha_1 + \alpha_0 \alpha_1^* \quad (2.95)$$

$$\langle \sigma_y \rangle = i\alpha_0 \alpha_1^* - i\alpha_0^* \alpha_1 \quad (2.96)$$

$$\langle \sigma_z \rangle = \alpha_0 \alpha_0^* - \alpha_1 \alpha_1^*. \quad (2.97)$$

Each statistical repetition of the experiment gives one output state vector which is depicted as a yellow dot on the Bloch sphere and the ensemble average is represented by a yellow arrow.

We visualize the QFT output for an example with eight qubits starting in the initial state  $|00000000\rangle$ . The state of the final qubit register can be seen in figure 2.29. The error parameters for this run were  $\sigma = 10^{-2}$  and  $p = 5 \cdot 10^{-3}$ , and  $m = 1000000$  statistical iterations were done. For illustration purposes only each 10<sup>th</sup> result is plotted. With these error parameters the overall error norm is quite small as can be seen in figure 2.27(a), so that the final vector is not supposed to have a large deviation from the ideal, error-free case. Starting with all qubits in the zero state, the ideal QFT transforms this input state to the

uniform superposition state, i.e., the expectation values for each qubit are supposed to be  $\langle \sigma_x \rangle = 1$ ,  $\langle \sigma_y \rangle = 0$  and  $\langle \sigma_z \rangle = 0$ . Figure 2.29 shows, that the ensemble averages all lie in the direction of the positive  $x$ -axis (with minor deviations). A more detailed look at the single experiments reveals how the yellow patches and structured patterns can occur. Let us retrace for example the transformation of the two lowest qubits to understand the evolution of the regular patterns around the equator. As described in figure 2.4 the first qubit (i.e. the last before the final swap operations) starts from  $|0\rangle$  and undergoes a Hadamard transform. That is, starting from  $(0, 0, 1)$  the Hadamard gate transforms the state to  $(1, 0, 0)$ . The subsequent decoherence operation ( $\in \{I, \sigma_x, \sigma_y, \sigma_z\}$ ) either leaves the state untouched ( $I$  or  $\sigma_x$ ) or transforms it into  $2^{-1/2}(|0\rangle - |1\rangle)$  ( $\sigma_y$  or  $\sigma_z$ ). In the unit sphere this corresponds to staying in  $(1, 0, 0)$  or a flip to  $(-1, 0, 0)$ . That is why there are two yellow patches on opposite sites on the equator of the sphere. Qubit number two is additionally subject to a phase rotation about the angle  $2\pi/2^2 = \pi/2$ . This can additionally lead to the vectors  $(0, 1, 0)$  and  $(0, -1, 0)$ . Possible subsequent decoherence errors don't move the vector out of the set of already mentioned vectors. By continuing this observation for the higher qubits, where the Hadamard operation on qubit  $q$  is followed by  $n - q$  controlled phase rotations, the doubling of the patches around the equator and the formation of the stable symmetric structure is understandable.

The operational errors, i.e. unitary over-rotations, cause a smearing of the resulting clusters, where the scattering grows with larger  $\sigma$ . Subsequent noisy operations can propagate those errors further.

Starting with the sixth qubit the possible locations of state vector endpoints around the equator from different error paths may overlap, so that it comes to an interference effect, where certain endpoint probabilities are more pronounced and one can see specific patterns emerging as for example clearly visible for the last qubit.

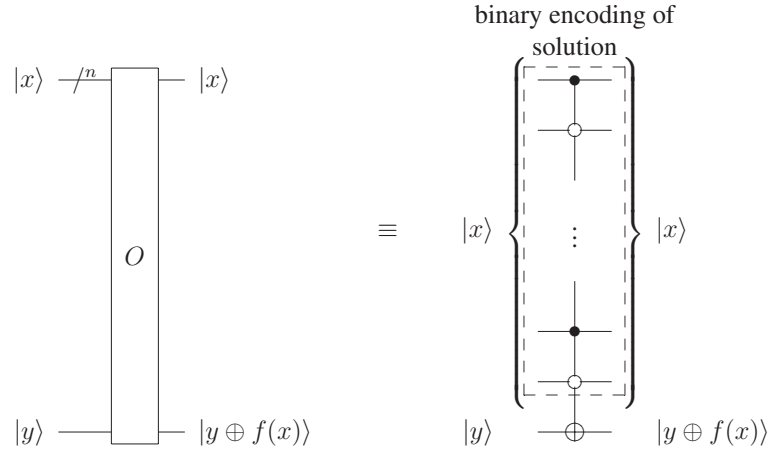
In summary, we can say that the quantum Fourier transform, due to its regular structure, shows an inherent robustness against decoherence and operational errors (compared to Grover's algorithm; see section 2.2.4). We found a threshold behavior for the magnitude of operational errors, which shows no significant dependency on the system size. For decoherence errors this is not the case and the conclusion is that doubling the system size requires a decrement of one order of magnitude in the gate error probability to keep a constant error norm.

### 2.2.4 Grover's Search Algorithm

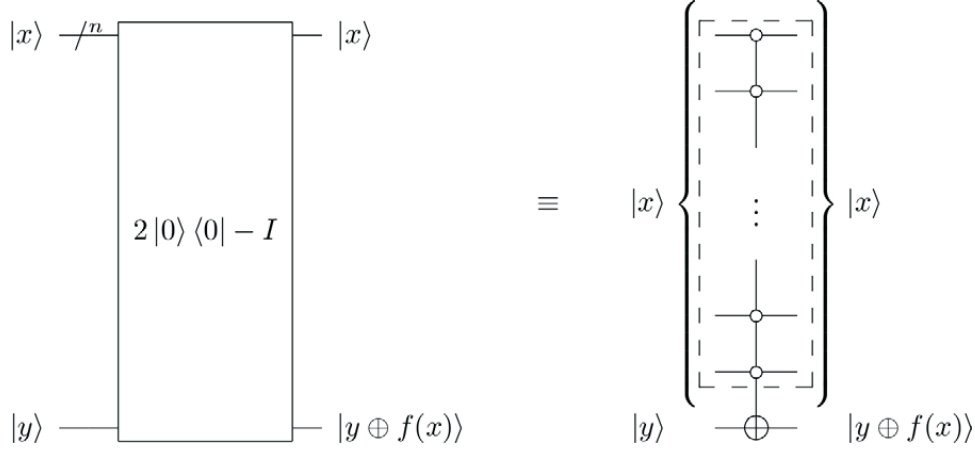
The ideal version of Grover's search algorithm is described in detail in section 2.1.2. We assess Grover's algorithm in case of non-ideal operations and under the influence of decoherence. Here we concentrate on the case of a single solution to the search problem ( $M = 1$ ). We use the error model described in section 2.2.1.

Although the oracle function is usually implemented as a black box, we describe our implementation of the oracle call in more detail, because the way of its implementation is essential for the results produced. The phase inversion of the solution element and the inversion about the mean is realized by using an oracle qubit as described in section 2.1.2, equations (2.32) – (2.34). The oracle function (equation (2.32)) can be implemented with a generalization of the  $C^n$ NOT gate, that uses both control-on-one and control-on-zero for the control qubits (see appendix A). The control qubits represent the binary encoding of the solution element (see figure 2.30). In principle, the  $C^n$ NOT gate could be build from  $\mathcal{O}(n)$  elementary gates (see [Nielsen and Chuang, 2000]), but here we use a direct implementation of the  $C^n$ NOT gate. The NOT operation on the target qubit is error-prone, whereas the control qubits are assumed to be error-free. That means that our black box oracle flips the phase of the solution element by conditionally flipping the oracle qubit. Of course, this sort of hard-coded oracle is not applicable for solving real problems, but here this gives us an efficient implementation of a black box oracle, which is suitable for our analyses. A proof that an efficient implementation can always be constructed can be found in [Nielsen and Chuang, 2000].

The operator  $(2|0\rangle\langle 0| - I)$ , which does the inversion about the mean (see equation (2.35)),



**Figure 2.30** – Implementation of the black box oracle for Grover's search algorithm to find the single solution of the search problem. It uses a generalized  $C^n$ NOT gate.



**Figure 2.31** – Implementation of the operator  $(2|0\rangle\langle 0| - I)$  in Grover's search algorithm using a generalized  $C^n$ NOT gate.

can be implemented (up to a global phase of  $-1$ , which is irrelevant) similarly to the oracle call with the binary encoding of the  $C^n$ NOT set to the solution element  $|0\rangle$  as shown in figure 2.31.

The simulation results for the system sizes<sup>12</sup>  $n' = 8 + 1$ ,  $n' = 16 + 1$  and  $n' = 23 + 1$  of the error-prone Grover algorithm are plotted in figure 2.32.<sup>13</sup> We are evaluating the square root fidelity depending on the number of Grover iterations. As explained in section 2.1.2, the optimal number of Grover iterations for the error free case is given by equation (2.43). For the system sizes  $n' = 8 + 1$ ,  $n' = 16 + 1$  and  $n' = 23 + 1$  the ideal number of iterations  $l$ , i.e., where the first maximum is expected, are  $l = 12$ ,  $l = 201$  and  $l = 2274$ . The number of statistical iterations has been chosen according to the computational effort, so that for the  $n' = 8 + 1$  qubit system  $m = 10^5$  iterations were done, for the  $n' = 16 + 1$  system  $m = 10^4$  and for the  $n' = 23 + 1$  system  $m = 100$ . The simulations were run on the JUMP system with a 256 processors and on JUBL<sup>14</sup> with a partition size of 2048 processors.

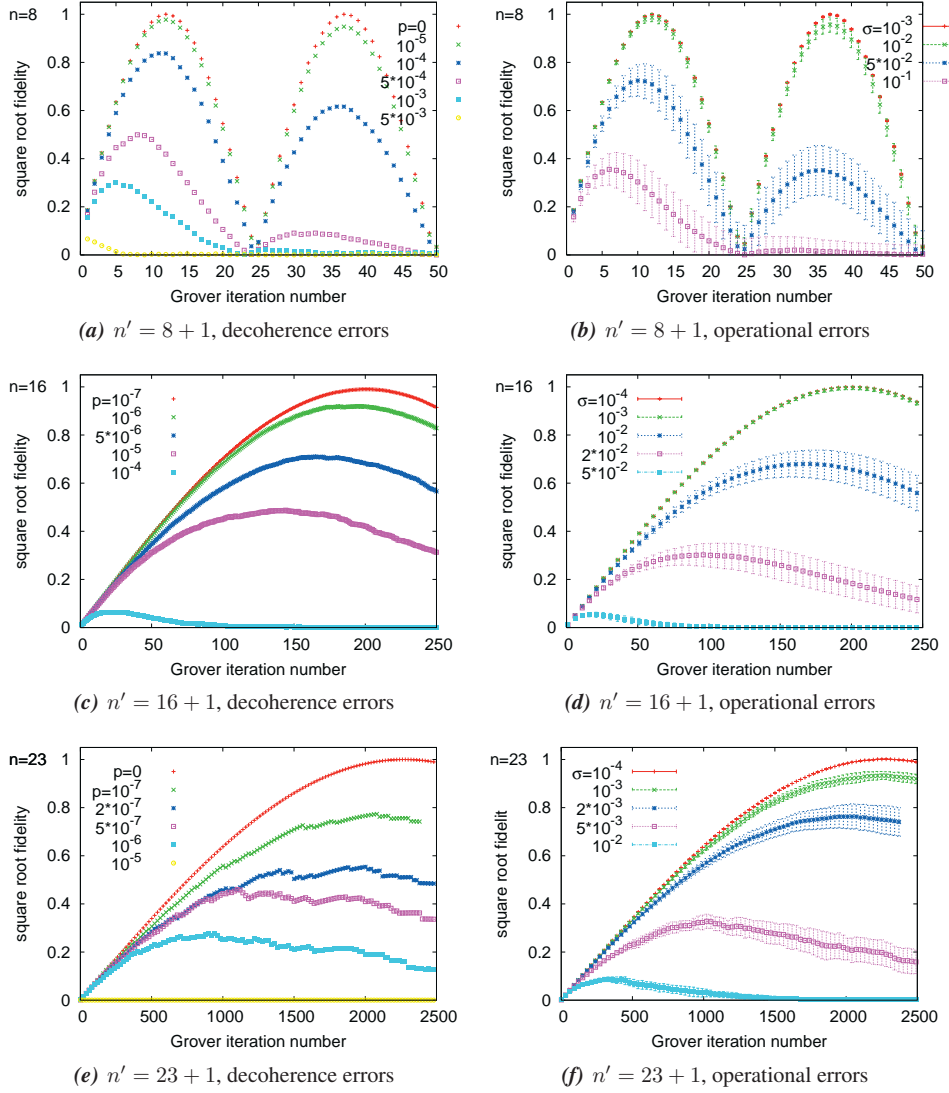
The maximum probability of finding the right element decreases with larger errors, while the number of Grover iterations necessary to reach the first maximum shifts to smaller values with increasing error rates. The left shift of the maximum is due to the exponential damping of the oscillating amplitude [Salas, 2008]. This means that less computational effort is needed to reach the maximum. Nevertheless, a careful examination shows that the

<sup>12</sup>The size  $n'$  of the overall system is written as  $n + 1$  indicating the additional oracle qubit. The number of database qubits  $n$  defines the size of the search space  $N = 2^n$ .

<sup>13</sup>The size  $n' = 23 + 1$  was chosen as the largest system size, because the computational effort to simulate that system already required  $35 \cdot 10^6$  quantum operations to collect statistics from  $m = 100$  runs.

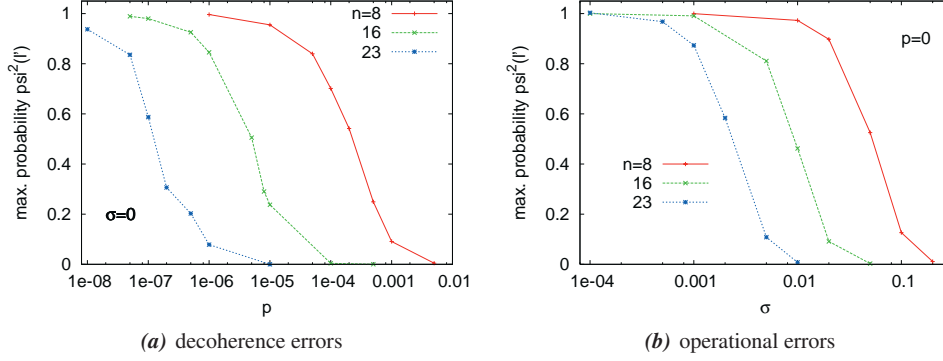
<sup>14</sup>Jülich BlueGene/L: This was an 8 rack system with a total of 8192 compute nodes, each with dual PowerPC 440 700 MHz processors. The peak performance was 45.8 Teraflop/s and it had an aggregated memory of 4.1 TB. Meanwhile, JUBL has been replaced with the Jülich BlueGene/P (JUGENE).





**Figure 2.32** – Simulation results of Grover's algorithm with decoherence errors (left column) and operational errors (right column).

decrease of the amplitude outweighs the shift of the maximum to lower iteration numbers [Peschina, 2008]. It is not possible to gain a benefit by increasing the level of noise. However, if a system is suffering from noise, stopping at lower Grover iteration numbers can save computational effort as well as increase the probability to find the correct solution.



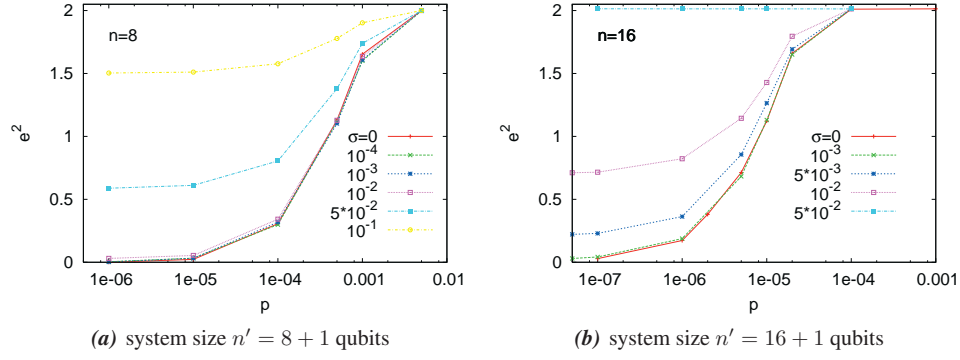
**Figure 2.33** – Maximum probability of finding the correct database entry with Grover’s algorithm in the presence of decoherence (a) or operational errors (b). The probability is evaluated at the position  $l'$  of the actual respective maximum (see figure 2.32). Lines are guides to the eye.

The sensitivity to decoherence and operational errors are very different in the chosen parameterization, but with a well-chosen parameter range both sorts of errors yield a qualitatively similar behavior. This is not surprising, since both error model approaches can be converted into each other for a single qubit (see section B).

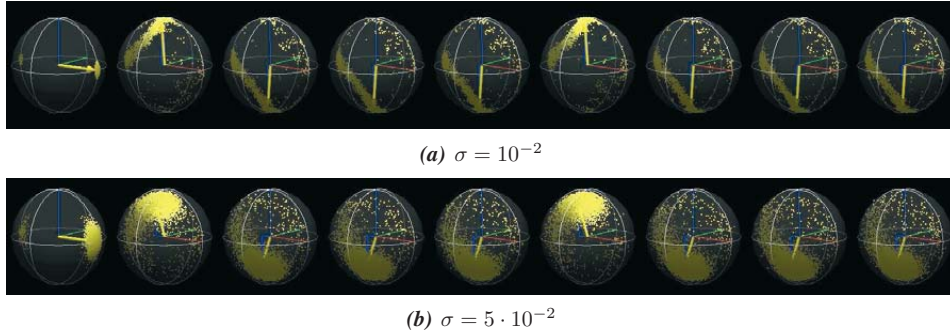
The probability of finding the correct database entry for different system sizes in case of decoherence errors or operational errors can be found in figure 2.33. Grover’s algorithm exhibits a stronger system size dependency than the QFT algorithm. As figure 2.33(a) indicates, a doubling of the system size requires a reduction of the single qubit decoherence probability of almost two orders of magnitude to keep a constant probability of success of the Grover search.

To compare the error sensitivity of Grover’s algorithm to that of the quantum Fourier transform we examine the error norm (equation (2.92)) for the system sizes  $n' = 8 + 1$  and  $n' = 16 + 1$  qubits (figure 2.34). In contrast to the quantum Fourier transform where the threshold behavior for operational errors is nearly independent of the system size (section 2.2.3, figure 2.27), here we see an explicit dependency of this threshold, which decreases significantly with system size. For values below this threshold operational errors have only marginal impact on the error norm, while for values above this threshold the error norm increases dramatically.

A visualization of the effects of both sorts of errors is depicted in figure 2.35. It clearly indicates the qualitatively different behavior of Grover’s algorithm suffering from operational errors above and below the numerically derived threshold of  $\sigma = 10^{-2}$  for  $n' = 8 + 1$  qubits. Each yellow dot is the result of a single run of the stochastic error model and the yellow arrow is the ensemble average. The leftmost qubit represents the ancilla qubit, which should be in the state of an equal superposition throughout the whole runtime of the algorithm.



**Figure 2.34** – Error norm in Grover’s algorithm in the presence of both decoherence and operational errors. There is a system size dependent threshold, which is  $\sigma \approx 10^{-2}$  for the  $n' = 8 + 1$  system and  $\sigma \approx 10^{-3}$  for  $n' = 16 + 1$  qubits. Lines are guides to the eye.



**Figure 2.35** – Grover’s algorithm with  $n' = 8 + 1$  qubits. Each yellow dot represents the final state of Grover’s algorithm after  $l=12$  Grover iterations. The yellow arrow represents the ensemble average of all runs with our statistical error model. The single qubit decoherence probability is set to  $p = 10^{-4}$  and the standard deviation of the operational errors are chosen to be  $\sigma = 10^{-2}$  (a) and  $\sigma = 5 \cdot 10^{-2}$  (b). The difference in error norm between these two cases can be found in figure 2.34(a), which already indicates a qualitatively different behavior. The leftmost qubit depicts the ancillary qubit used in Grover’s algorithm, while the other eight qubits encode the solution database element, which in this case has been chosen to be  $k = 17$ . The least significant bit is on the left. Note that the axes have been inverted in this picture, so that  $|0\rangle$  points down and  $|1\rangle$  points up. The yellow arrows indicate that in the first case the binary encoding of the solution is found with only minor deviations (a), whereas in the second case a clear deviation from the expected ideal solution can be seen (b).

The other eight qubits should encode the solution element  $k = 17$  of the database search, beginning with the least significant bit on the left. While the single qubit decoherence rate  $p = 10^{-4}$  is kept constant, a value of  $\sigma = 10^{-2}$  (figure 2.35(a)) leads to end vectors that mostly lie approximately within a plane dominated by decoherence errors, whereas a value of  $\sigma = 5 \cdot 10^{-2}$  exhibits a different behavior. Here the smearing due to operational errors dominates. Let us also emphasize the role of the ancilla qubit here. The ancilla qubit should be in the state  $2^{-1/2}(|0\rangle - |1\rangle)$  during the execution of the algorithm. Any deviation from this state will lead to an erroneous phase inversion of the solution element as well as an erroneous phase inversion about the mean (see section 2.1.2.2). Therefore the ancilla qubit is especially important for the correct operation of Grover's algorithm and in section 2.3 we analyze the case where the ancilla qubit is stabilized using a quantum error correction code.

The analysis of Grover's algorithm under noisy conditions shows that this algorithm is more fragile towards decoherence errors as well as operational errors than for example the quantum Fourier transform for comparable system sizes. Especially the threshold for operational errors is clearly dependent on the system size, so that a successful run of Grover's algorithm requires an extraordinarily good control of operational errors. Nevertheless, a non-perfect run (within certain error ranges) can still give acceptable results with high probability if the damping and therefore the shift of the maximum amplitude of the solution is taken into account. With our simulations we can quantify this resulting shift and therefore help to save computational effort while increasing the probability of finding the right solution for a noisy system.

## 2.3 Simulation of Quantum Error Correction

Building upon the algorithmic foundations of [Shor, 1994], [Grover, 1996] and many others it has been proven that quantum computers can in principle solve certain types of problems more efficiently than classical computers – neglecting decoherence phenomena and operational imprecisions. Yet, in reality every quantum computation device will suffer from decoherence as well as operational imprecisions. In section 2.2 we have analyzed the robustness of the most important algorithms against both sorts of errors and we can quantify the decrease in fidelity for certain amounts of noise. The question arises, if a practical quantum computation device can be built to overcome these inherent problems.

After some critical objections that the power of quantum computers cannot be harnessed for realistic error-prone devices [Landauer, 1995; Unruh, 1995], the discovery of quantum error correction schemes [Shor, 1995; Steane, 1996a,b] showed a way to overcome these objections and the future prospects for quantum computation gained a tremendous boost. Nevertheless, staying in the quantum circuit model, the encoding and recovery circuits are again inevitably suffering from errors themselves, so that a careful evaluation of error propagation is necessary. The combination of quantum error correction with fault-tolerant state recovery, fault-tolerant encoding of quantum logic operations and the principle of concatenation of quantum error correction codes culminates in the accuracy threshold theorem [Aharonov and Ben-Or, 1996]. This proves that arbitrarily long quantum calculations are possible if the single qubit error rate is below a certain threshold. This promising idea led to comprehensive research and development striving for a working scalable quantum computer.

The advances in the field of *fault-tolerant* quantum error correction led to the insight, that there is no fundamental barrier to realizing a large-scale quantum computer, although practical difficulties remain even today. Since quantum error correction and especially fault-tolerant quantum error correction need much more resources than simple error-correction-free quantum computations, it seems unlikely that fault-tolerant quantum error correction will be implemented for the first generations of quantum computers. But quantum error correction will play an important role for the long term storage of quantum information and for sending quantum information over noisy channels. One can be sure that future large-scale quantum computers will incorporate some sort of quantum error correction [Preskill, 1998]. Thus, modelling the effectiveness of quantum error correction codes is an important task for evaluating future quantum computing architectures.

The standard fault-tolerant theory of quantum error correction gives proofs for the existence of such a threshold, but the estimations for such a threshold range from  $10^{-7}$  to  $10^{-2}$ , depending on more or less realistic assumptions<sup>15</sup> [Zalka, 1996; Knill et al., 1996, 1998a; Steane, 2003; Aliferis et al., 2005; Knill, 2005; Reichardt, 2006].

---

<sup>15</sup>For example, the threshold of  $10^{-2}$  [Knill, 2005] requires about  $10^6$  physical qubits to encode a single logical qubit.

We answer the question for a realistic number of available qubits (i.e. a single level of concatenation; see section 2.3.3) and give a numerical threshold for the success of quantum error correction within the limits of our error model. Additionally, we quantify the effects of fighting operational errors within this quantum error correction framework. Standard fault-tolerant theory does not provide estimates for these unitary over-rotations, which can add up to significantly larger errors throughout the course of any quantum algorithm. With our simulations we can clear up doubts about the efficiency of Shor's algorithm in the presence of operational imprecisions formulated by [Hill and Viamontes, 2008]. We show that fault-tolerant quantum error correction is especially well suited for the correction of over-rotations. Usually no statement is made in the literature about the combination of both error types. Our simulations also provide the analysis of the interplay between both sources of error, idling qubits suffering from decoherence and noisy logic gate operations. Our numerical thresholds refer to actual solutions of practical problems and substantiate the existence proofs of the analytical works [Aharonov and Ben-Or, 1996; Knill et al., 1996; Kitaev, 1997a].

### 2.3.1 Quantum Error Correction Codes

Error correction in general is accomplished by redundantly encoding information. The simplest classical example is duplicating the information several times and recover any error by doing a majority voting. However, in the quantum realm this approach is unfeasible. The first obstacle is the no-cloning theorem [Wootters and Zurek, 1982], which states that in the quantum world it is not possible to make a perfect copy of an arbitrary unknown quantum state. Secondly, a straightforward evaluation of the majority voting is not possible, because a direct measurement would destroy any coherent quantum superposition. Finally, classical error correction has to deal with a single type of error only, namely a flip of a bit, whereas quantum states can suffer from a continuum of possible errors, that can even add up over time. That renders classical error correction techniques useless. However Shor [Shor, 1995] and Steane [Steane, 1996a,b] discovered a way to overcome these objections. The key idea is to encode a quantum state in a highly entangled state of additional supporting qubits. Thus, a small subspace of the system's Hilbert space is defined as the code subspace. This is chosen such that possible errors move the code subspace to mutually orthogonal error subspaces of the system. To avoid collapse of the quantum superposition by measurement operations it is necessary to extract the error information, that indicates a potential error, by partial measurement. The measurement result is called the error syndrome and gives information about the error only, without revealing information about the data itself. Linear combinations of correctable error are also correctable in the sense that the syndrome measurement projects the state into a well defined error subspace which can then be corrected by applying the appropriate unitary transformation which reverses the effect of the error. It is fundamental to quantum error correction that the ability to correct a discrete set of errors suffices to correct a much larger, even continuous class of errors.

Over time many different quantum error correction codes (QECC) had been developed. They are classified as  $[[n, k, d]]$  QECCs where  $k$  logical qubits are encoded in  $n$  physical qubits protecting against errors of distance  $d$ . The Hamming distance  $d$  comes from classical coding theory and states that going from any codeword in the code to any other codeword requires a flip of at least  $d$  bits. A linear code with distance of at least  $2t + 1$  can correct errors on up to  $t$  bits [Nielsen and Chuang, 2000].

We will concentrate on codes with  $k = 1$  and distance  $d = 3$ , which can correct an arbitrary error on a single logical qubit. An analysis of the performance of higher distance codes can be found in [Steane, 2003]. The first QECC that we describe is Shor's 9-qubit code [Shor, 1995], which first showed a way out of the conundrum of effective quantum error correction. Even more important is Steane's 7-qubit code [Steane, 1996a], because it plays a role in fault-tolerant quantum computation. It belongs to the most important class of codes, the so called stabilizer codes [Gottesman, 1997] (see section C) which are generalizations of the CSS codes (named after Calderbank, Shor and Steane) [Calderbank and Shor, 1996]. Another important quantum error correction code is Laflamme's 5-qubit code [Laflamme et al., 1996; DiVincenzo and Shor, 1996], which uses the smallest possible number of qubits to protect against any single qubit error.

A protection against a single qubit error means that we can enhance the fidelity  $F$  of an unknown quantum state that suffers from a single qubit error with probability  $p$  from  $F = 1 - p$  to  $F' = 1 - \mathcal{O}(p^2)$ , assuming uncorrelated errors between qubits of an encoded block and more importantly, perfect encoding, decoding and recovery operations. In section 2.3.3 we will see, how the second assumption can be relaxed by going to fault-tolerant quantum error correction and how it is possible to push the fidelity limit by using concatenated codes.

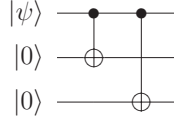
### 2.3.1.1 Shor's 9-Qubit Quantum Error Correction Code

In 1995 Shor devised the first quantum error correction code [Shor, 1995] that circumvents the obstacles of error correction applied to quantum states. The main building block of Shor's 9-qubit code is the 3-qubit bit flip code which encodes a logical qubit using three qubits. The logical qubit can be protected against a single bit flip  $\sigma_x$ . An arbitrary single qubit state is encoded according to

$$\alpha|0\rangle + \beta|1\rangle \rightarrow \alpha|0_L\rangle + \beta|1_L\rangle = \alpha|000\rangle + \beta|111\rangle \quad (2.98)$$

where  $|0_L\rangle$  and  $|1_L\rangle$  denote the logical states instead of the physical ones. A circuit for doing this encoding is depicted in figure 2.36.

If one qubit of the encoded system is flipped accidentally, this can be detected in a first step and corrected afterwards by flipping back the corresponding qubit. For the detection, a measurement has to be done that indicates what error, if any, has occurred. The result of this measurement is called the error syndrome. The syndrome can be measured by observing



**Figure 2.36** – Encoding circuit for the 3-qubit bit flip code.

the following projection operators:

$$P_0 = |000\rangle\langle 000| + |111\rangle\langle 111|, \quad (2.99)$$

$$P_1 = |100\rangle\langle 100| + |011\rangle\langle 011|, \quad (2.100)$$

$$P_2 = |010\rangle\langle 010| + |101\rangle\langle 101|, \quad (2.101)$$

$$P_3 = |001\rangle\langle 001| + |110\rangle\langle 110|. \quad (2.102)$$

The expectation value

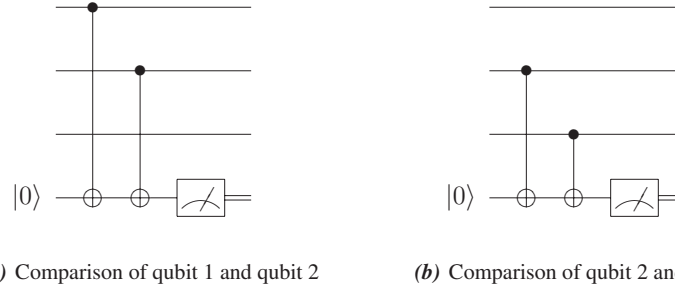
$$\langle \psi | P_i | \psi \rangle = 1, \quad i \in \{0, 1, 2, 3\} \quad (2.103)$$

indicates the location  $i$  of the bit which has possibly suffered a bit flip (with  $i = 0$  indicating the error-free case). This is independent of the actual state of the qubit, i.e. independent of the complex amplitudes  $\alpha$  and  $\beta$  describing the state. A generic feature of syndrome measurements is that no information about the protected state is obtained, i.e., one cannot infer anything about the values of  $\alpha$  and  $\beta$  from this measurement.

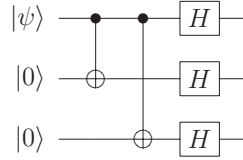
Since a measurement of the four projection operators defined in equations (2.99)–(2.102) is equivalent to measuring the two observables  $Z_1Z_2$  (which is  $Z \otimes Z \otimes \mathbb{1}$ ) and  $Z_2Z_3$  ( $\mathbb{1} \otimes Z \otimes Z$ )<sup>16</sup> [Nielsen and Chuang, 2000], often the syndrome is extracted by doing these two measurements: Each observable has two eigenvalues  $\pm 1$ , so that measuring both observables gives four possible outcomes, the syndromes. The measurement of one observable can be regarded as a measurement of the parity of the two corresponding qubits, i.e. a comparison to see if two qubits have the same value. The corresponding syndrome measurement circuit consists of two pairwise measurements of the parity using an additional ancilla qubit. (figure 2.37). An even parity of qubits 1 and 2 corresponds to a measurement result of 0 for the ancilla qubit (figure 2.37(a)) as well as an eigenvalue of  $+1$  for the observable  $Z_1Z_2$ . It tells us, that both qubits have the same value. An odd parity or a measurement result of 1 for the ancilla or an eigenvalue of  $-1$  for the observable  $Z_1Z_2$  shows that both qubits have different values. From the result of both measurements (figures 2.37(a) and 2.37(b)) one can easily deduce which qubit, if any, has suffered a bit flip: If the measurement result is 0 in both cases, no bit has been flipped, if the result of the first measurement (figure 2.37(a)) is 0 and the result of the second measurement (figure 2.37(b)) is 1, we know that the third qubit must have been flipped. In case the first measurement gives 1 and the second gives 0, we know that qubit 1 must have suffered a bit flip and if

<sup>16</sup> $Z_1Z_2$  and  $Z_2Z_3$  are arbitrarily chosen. Any other combination between different qubits, such as  $Z_1Z_2$  and  $Z_1Z_3$  or the combination  $Z_1Z_3$  and  $Z_2Z_3$  is also possible.





**Figure 2.37** – Syndrome measurement circuit for the 3-qubit bit flip code. Two measurements have to be done to locate the position of a potential bit flip. Each measurement compares the state of two qubits and involves an ancilla qubit that is prepared in the state  $|0\rangle$ . The combination of both measurement results (a) and (b) gives unique information about a potential single bit flip.



**Figure 2.38** – Encoding circuit for the 3-qubit phase flip code.

both measurement are 1, the second qubit has been flipped. Knowing which bit has flipped, a recovery operation can be done by flipping the same qubit back.

Phase flips can be dealt with by going to the rotated basis  $|+\rangle = 2^{-1/2}(|0\rangle + |1\rangle)$  and  $|-\rangle = 2^{-1/2}(|0\rangle - |1\rangle)$ . A phase flip takes  $|+\rangle$  to  $|-\rangle$  and vice versa, i.e., it acts like a bit flip with respect to the rotated basis. That means that qubits can be protected from phase flips by encoding them with the encoding circuit in figure 2.38.

Shor's 9-qubit code is just the concatenation of the bit flip with the phase flip code, so that the qubit is protected against both kinds of errors and therefore against arbitrary errors on a single qubit. The left side of figure 2.39 shows the encoding and the right side the decoding circuit.

In principle, the decoding circuit is just the encoding circuit run in reverse, but here we use a variant, where the recovery step is done implicitly during the decoding. This has the advantage that the circuits become simpler and less operations have to be done, thus reducing the error-proneness. The following example for the 3-qubit bit flip code explains why this procedure is justified: A general state  $\alpha|0\rangle + \beta|1\rangle$  is encoded into  $\alpha|000\rangle + \beta|111\rangle$ .

Consider the cases

$$\alpha |000\rangle + \beta |111\rangle \xrightarrow{\text{error}} \begin{cases} \alpha |000\rangle + \beta |111\rangle & \text{if no error has occurred} \\ \alpha |100\rangle + \beta |011\rangle & \text{if qubit 1 has flipped} \\ \alpha |010\rangle + \beta |101\rangle & \text{if qubit 2 has flipped} \\ \alpha |001\rangle + \beta |110\rangle & \text{if qubit 3 has flipped.} \end{cases} \quad (2.104)$$

The decoding circuit transforms these states into

$$\xrightarrow{\text{decode}} \begin{cases} \alpha |000\rangle + \beta |100\rangle = (\alpha |0\rangle + \beta |1\rangle) |00\rangle \\ \alpha |111\rangle + \beta |011\rangle = (\alpha |1\rangle + \beta |0\rangle) |11\rangle \\ \alpha |010\rangle + \beta |110\rangle = (\alpha |0\rangle + \beta |1\rangle) |10\rangle \\ \alpha |001\rangle + \beta |101\rangle = (\alpha |0\rangle + \beta |1\rangle) |01\rangle . \end{cases} \quad (2.105)$$

The Toffoli gate of the recovery step turns the possible states into

$$\xrightarrow{\text{recovery}} \begin{cases} (\alpha |0\rangle + \beta |1\rangle) |00\rangle \\ (\alpha |0\rangle + \beta |1\rangle) |11\rangle \\ (\alpha |0\rangle + \beta |1\rangle) |10\rangle \\ (\alpha |0\rangle + \beta |1\rangle) |01\rangle . \end{cases} \quad (2.106)$$

This means that the original state of the qubit has been restored. The only difference to the recovery after syndrome measurement is that the state of the additional qubits (qubits 2 and 3) is not necessarily restored to  $|00\rangle$ , but they can remain in any combination of basis states. Since their state is not of interest after the qubit has been decoded, they can be measured later on or reset prior to using them again.

The code space in Shor's 9-qubit code is spanned by the logical states

$$|0\rangle \rightarrow |0_L\rangle = \frac{1}{2\sqrt{2}} \left( |000\rangle + |111\rangle \right) \left( |000\rangle + |111\rangle \right) \left( |000\rangle + |111\rangle \right) \quad (2.107)$$

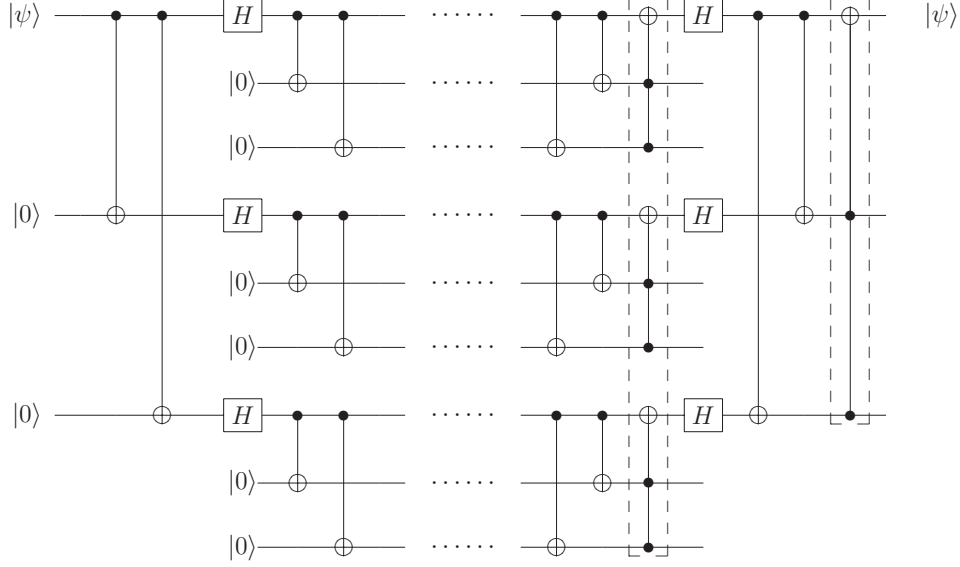
and

$$|1\rangle \rightarrow |1_L\rangle = \frac{1}{2\sqrt{2}} \left( |000\rangle - |111\rangle \right) \left( |000\rangle - |111\rangle \right) \left( |000\rangle - |111\rangle \right). \quad (2.108)$$

Altogether, Shor's 9-qubit code uses nine qubits for one logical qubit and protects that logical qubit against an arbitrary error on a single qubit.

### 2.3.1.2 Steane's 7-Qubit Quantum Error Correction Code

Steane's 7-qubit code [Steane, 1996a,b] is probably the most interesting quantum error correction code for protecting a single qubit against arbitrary errors, especially because it is



**Figure 2.39** – Encoding, recovery and decoding circuit for Shor’s 9-qubit code. The left part shows the encoding circuit, which is a concatenation of the 3-qubit bit flip code with the 3-qubit phase flip code. In principle, the decoding circuit is just the encoding circuit run in reverse. Here, we show a variant where the syndrome measurement and recovery step is done implicitly during decoding (dashed boxes).

well suited for fault-tolerant approaches, as we will see in section 2.3.3. For this reason we will study this particular code in more detail.

Steane’s 7-qubit code  $[[7, 1, 3]]$  is closely related to the classical  $[7, 4, 3]$  Hamming code [MacWilliams and Sloane, 1977]. For the understanding of Steane’s code it is helpful to have a look at the classical Hamming code first, which uses  $n = 7$  bits to encode  $k = 4$  bits of classical data. There are  $2^k$  strings of length  $n$  that are valid codewords. These codewords  $v$  satisfy

$$Hv = 0 \pmod{2}, \quad (2.109)$$

with  $H$  being the  $n - k$  by  $n$  parity check matrix

$$H = \begin{pmatrix} 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 0 & 1 & 1 & 0 & 0 & 1 & 1 \\ 1 & 0 & 1 & 0 & 1 & 0 & 1 \end{pmatrix}. \quad (2.110)$$

The code is defined to be the kernel of  $H$ , which must be  $k$ -dimensional. That is,  $H$  has three linearly independent rows and the kernel is spanned by four linearly independent  $n$ -element vectors.

The parity check matrix formulation offers an obvious way to apply error correction. It is now possible to determine the position of a possible error, i.e. a single bit flip at an unknown position, in a valid but unknown codeword. This is done by applying the parity check matrix to the  $n$ -bit string. Suppose an error occurs on the  $i^{\text{th}}$  bit of the string, i.e.

$$v' = v \oplus e_i, \quad (2.111)$$

with  $e_i$  being the unit vector with entry 1 in the  $i^{\text{th}}$  component and “ $\oplus$ ” denoting bitwise addition modulo 2. Applying the parity check matrix  $H$ ,

$$Hv' = H(v \oplus e_i) = He_i, \quad (2.112)$$

gives the  $i^{\text{th}}$  column of the matrix, from which the error position can be inferred directly, as it gives the binary representation of  $i$ . Correcting the error is done by flipping back the  $i^{\text{th}}$  bit. Notice that the position of a possible error is revealed, but no information about the encoded data itself.

Steane’s  $[[7, 1, 3]]$  code is a generalization of this classical code to a quantum code (compare equation (2.110) and table C.5, p. 156). It uses the logical codewords

$$\begin{aligned} |0_L\rangle = \frac{1}{\sqrt{8}} (&|0000000\rangle + |0001111\rangle + |0110011\rangle + |0111100\rangle \\ &+ |1010101\rangle + |1011010\rangle + |1100110\rangle + |1101001\rangle), \end{aligned} \quad (2.113)$$

the superposition of all even weight Hamming codewords, and

$$\begin{aligned} |1_L\rangle = \frac{1}{\sqrt{8}} (&|1111111\rangle + |1110000\rangle + |1001100\rangle + |1000011\rangle \\ &+ |0101010\rangle + |0100101\rangle + |0011001\rangle + |0010110\rangle), \end{aligned} \quad (2.114)$$

the superposition of all odd weight Hamming codewords.

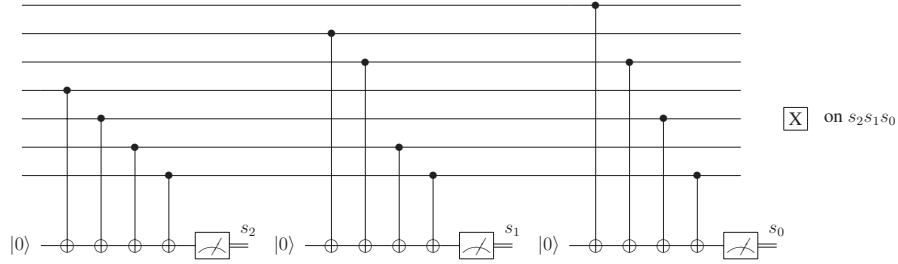
From the parity check matrix (equation (2.110)) the (non-fault-tolerant) syndrome measurement circuit can immediately be derived (figure 2.40).

In Steane’s code the recovery is done in two steps. Figure 2.40 shows the bit flip correction part of Steane’s recovery circuit. The whole circuit has to be applied a second time in the rotated basis, i.e. enclosed in Hadamard operations on all qubits at the beginning and at the end of the circuit. It is easy to check, that Hadamard operations applied to all qubits also rotate the logical basis states,

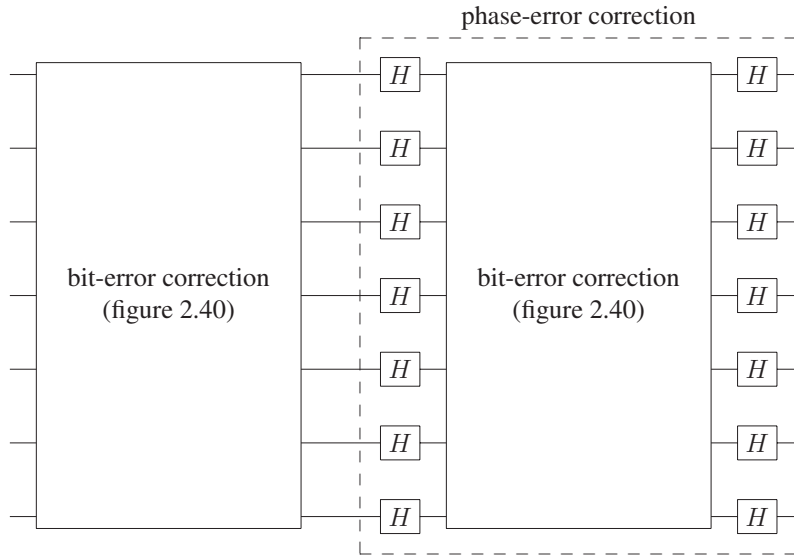
$$|0_L\rangle \xrightarrow{H^{\otimes 7}} \frac{1}{\sqrt{2}}(|0_L\rangle + |1_L\rangle), \quad (2.115)$$

and

$$|1_L\rangle \xrightarrow{H^{\otimes 7}} \frac{1}{\sqrt{2}}(|0_L\rangle - |1_L\rangle). \quad (2.116)$$



**Figure 2.40** – Syndrome measurement circuit for Steane’s 7-qubit code. Additional ancilla qubits are needed to extract the syndrome information. The syndrome  $s_2s_1s_0$  gives the location of a possible bit flip error in binary notation. The recovery operation would be a bit flip on the  $s_2s_1s_0^{\text{th}}$  qubit.

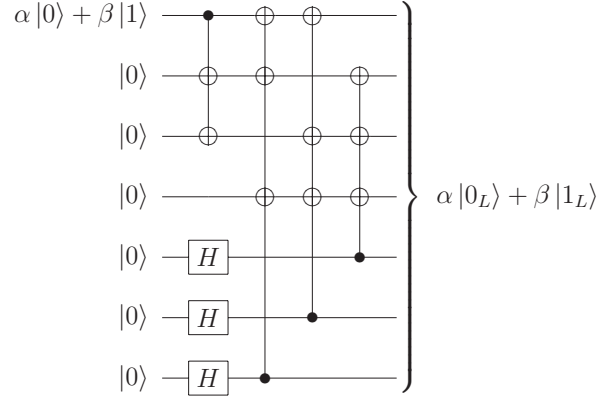


**Figure 2.41** – Syndrome measurement and recovery in Steane’s code is done in two steps. First bit flip errors are corrected, then phase flip errors and therefore possible combinations thereof.

Therefore, a recovery of bit flip errors in the rotated basis does the same as a recovery of phase flip errors in the original basis. A full recovery step is depicted in figure 2.41.<sup>17</sup>

An unknown quantum state  $|\psi\rangle = \alpha|0\rangle + \beta|1\rangle$  can be encoded by using the circuit shown in figure 2.42. The operation of the encoding circuit can be understood by using an alternative

<sup>17</sup>Actually, this circuit does not only correct an arbitrary error on a single qubit, but it can also recover one bit flip error and one phase flip error on different qubits.



**Figure 2.42** – Encoding circuit for Steane’s 7-qubit code. The decoding circuit is just the encoding circuit in reverse.

expression of the Hamming parity check matrix (equation (2.110)) with a re-ordering of the columns,

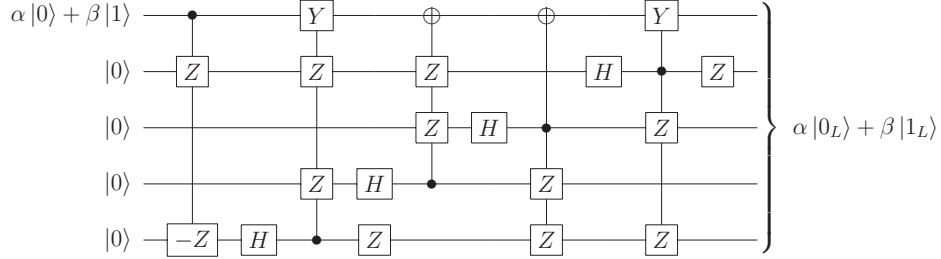
$$H' = \begin{pmatrix} 1 & 0 & 0 & 1 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 & 1 & 0 & 1 \\ 0 & 0 & 1 & 1 & 1 & 1 & 0 \end{pmatrix}. \quad (2.117)$$

The logical zero (equation (2.113)) is just the space spanned by the rows of  $H$  and in the representation of  $H'$  the first three qubits contain the information in the subcode. The additional four qubits are parity qubits, which add the required redundancy to protect the state from errors. The first and second CNOT gates create the state  $\alpha|0000000\rangle + \beta|0000111\rangle$ , the Hadamard rotations create the equal superposition of all eight possible values for the highest qubits and the following CNOT gates set the parity bits according to the parity check matrix  $H'$ .<sup>18</sup> The decoding circuit is just the encoding circuit run in reverse.

### 2.3.1.3 5-Qubit Quantum Error Correction Code

It can be shown that using five qubits to encode a quantum state is the minimal number of qubits to protect a quantum state against a single qubit error [Nielsen and Chuang, 2000].

<sup>18</sup>Actually, this creates logical states, where the logical codewords are not exactly those given by equations (2.113) and (2.114), but the qubits are numbered in reverse order, i.e., the least significant bit is on the right and the most significant bit on the left. This is just another convention, e.g. used in [Gottesman, 1997]. Still, the logical zero is the superposition of all even weight codewords and the logical one consists of all odd weight codewords.



**Figure 2.43** – Encoding circuit for the 5-qubit code. The decoding circuit is just the encoding circuit in reverse.

The logical codewords of the five qubit code [Laflamme et al., 1996; DiVincenzo and Shor, 1996] are

$$\begin{aligned}
 |0_L\rangle = \frac{1}{4} & (|00000\rangle + |10010\rangle + |01001\rangle + |10100\rangle \\
 & + |01010\rangle - |11011\rangle - |00110\rangle - |11000\rangle \\
 & - |11101\rangle - |00011\rangle - |11110\rangle - |01111\rangle \\
 & - |10001\rangle - |01100\rangle - |10111\rangle + |00101\rangle )
 \end{aligned} \tag{2.118}$$

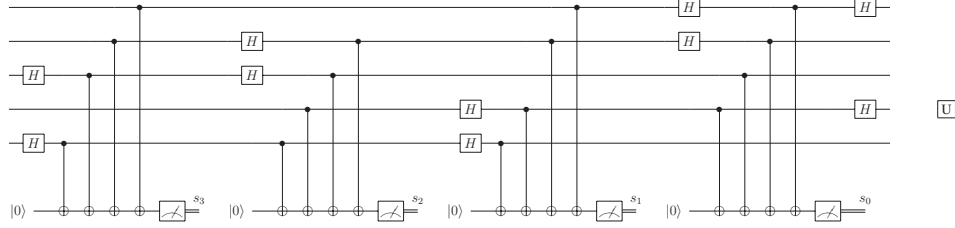
and

$$\begin{aligned}
 |1_L\rangle = \frac{1}{4} & (|11111\rangle + |01101\rangle + |10110\rangle + |01011\rangle \\
 & + |10101\rangle - |00100\rangle - |11001\rangle - |00111\rangle \\
 & - |00010\rangle - |11100\rangle - |00001\rangle - |10000\rangle \\
 & - |01110\rangle - |10011\rangle - |01000\rangle + |11010\rangle ).
 \end{aligned} \tag{2.119}$$

A possible encoding circuit is shown in figure 2.43 [Niwa et al., 2002].

A (non-fault-tolerant) syndrome measurement and recovery circuit is shown in figure 2.44. The recovery circuit can be derived directly from the stabilizers<sup>19</sup> of the 5-qubit code (see appendix C, table C.6). We use the same circuit as in [Niwa et al., 2002]. From the circuit it can be deduced that they chose another representation of the stabilizer with the generators found in table 2.1. The necessary correction operation, denoted by  $U$  in figure 2.44, can be determined from those generators. The four syndrome bits uniquely define what kind of error has occurred, either a bit flip, a phase flip, or a combined bit and phase flip, and on which of the five qubits the error has happened.

<sup>19</sup>A brief introduction into the stabilizer formalism [Gottesman, 1997] is given in appendix C.



**Figure 2.44** – Syndrome measurement and recovery circuit for the 5-qubit code (non-fault-tolerant). The recovery operation  $U$  reverses a possible error; a bit flip, a phase flip or a combined bit and phase flip on one of the qubits, depending on the measured syndrome. The operation  $U$  can be derived from table 2.1.

syndrome bits \ qubit		qubit				
		1	2	3	4	5
	$s_3$	Z	Z	X	I	X
	$s_2$	I	X	Z	Z	X
	$s_1$	Z	X	I	X	Z
	$s_0$	X	Z	Z	X	I

**Table 2.1** – Generators of the 5-qubit code and corresponding syndromes. If all syndrome bits are zero, no error has occurred. If a bit flip error has occurred on one of the qubits, the syndrome bits of the corresponding column will be set where an Z is found. For a phase flip the same applies to the operator X. This might seem counterintuitive at first, but remember that measuring the observable Z detects bit flips and measuring the observable X detects phase flips (see figure 2.37). Combinations of bit and phase flips are also possible. For example, if qubit 3 was affected by a combined bit and phase flip, the syndrome measurement would give the syndrome  $s_3s_2s_1s_0 = 1101$  as a binary number. A measured syndrome of  $s_3s_2s_1s_0 = 0110$ , for instance, would indicate a phase flip on qubit 2, whereas a bit flip on qubit 2 would result in the syndrome  $s_3s_2s_1s_0 = 1001$ . The recovery operation  $U$  flips the erroneous qubit back by doing the same flip operation a second time.



### 2.3.2 Simulation Results of Ideal and Error-Prone Quantum Error Correction

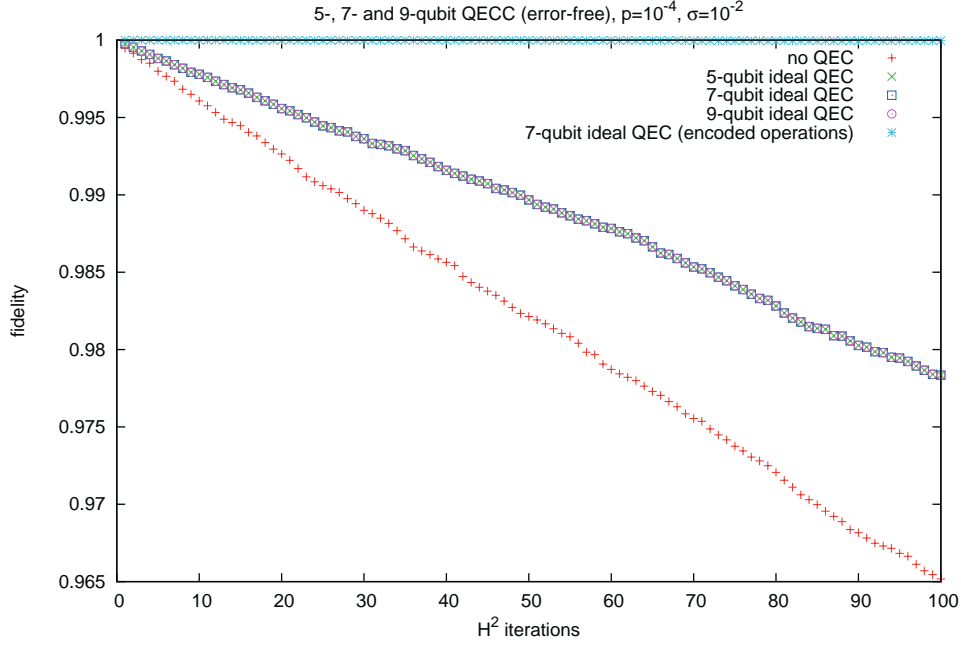
As a first step we implemented the error correction schemes described in section 2.3.1. We compare the performance of these codes with that of the unencoded case. The first checks of the simulation code were done by running the simulations with ideal error-free quantum error correction, i.e., the original quantum circuit is prone to error, while the quantum error correction circuits are assumed to be error-free. The quantum algorithm considered in this case is the  $H^{2k}$ -algorithm for which a robustness analysis for the unprotected case has already been done (section 2.2.2).

From theoretical considerations (section 2.3.1) we already know that we can correct any single qubit error within a block if we assume the error correction circuit to be error free. Therefore, a quantum error correction step will never degrade the fidelity of a quantum state. The first analysis is part of the verification of the correct implementation of the different quantum error correction schemes. We just give a brief summary for the error-free quantum error correction and will concentrate our analyses on the realistic case, where the error correction circuits are also prone to error (section 2.3.3).

Figure 2.45 shows an example of error-free quantum error correction. The length of the quantum algorithm has been chosen to be 100  $H^2$  operations. The number of statistical iterations is  $m = 10^5$  for the unencoded case and  $m = 10^3$  for the encoded runs. Here we show only a single parameter set with  $p = 10^{-4}$  and  $\sigma = 10^{-2}$ .

Examining the 5-, 7-, and 9-qubit quantum error correction codes, all these codes show the ability to protect against decoherence noise. For the protection of quantum memory, i.e. idling qubits, all codes are equally well suited (in the case of ideal quantum error correction). Yet, going to quantum computation requires the ability to do operations on encoded states, because even in the ideal case a decoding with subsequent gate operation and re-encoding leaves the qubit temporarily unprotected. Additionally, unwanted unitary over-rotations, cannot be detected and corrected, because an unwanted additional rotation angle on the temporarily unencoded qubit is indistinguishable from an intended rotation. Doing logical operations directly on encoded states requires additional considerations. The special structure of the 7-qubit quantum error correction code allows the realization of encoded operations (see section 2.3.3). With logical operations on encoded states, it is possible to distinguish the exact logical operation from an imprecise rotation. Therefore, the 7-qubit code is able to correct operational over-rotations, when using operations on encoded states.

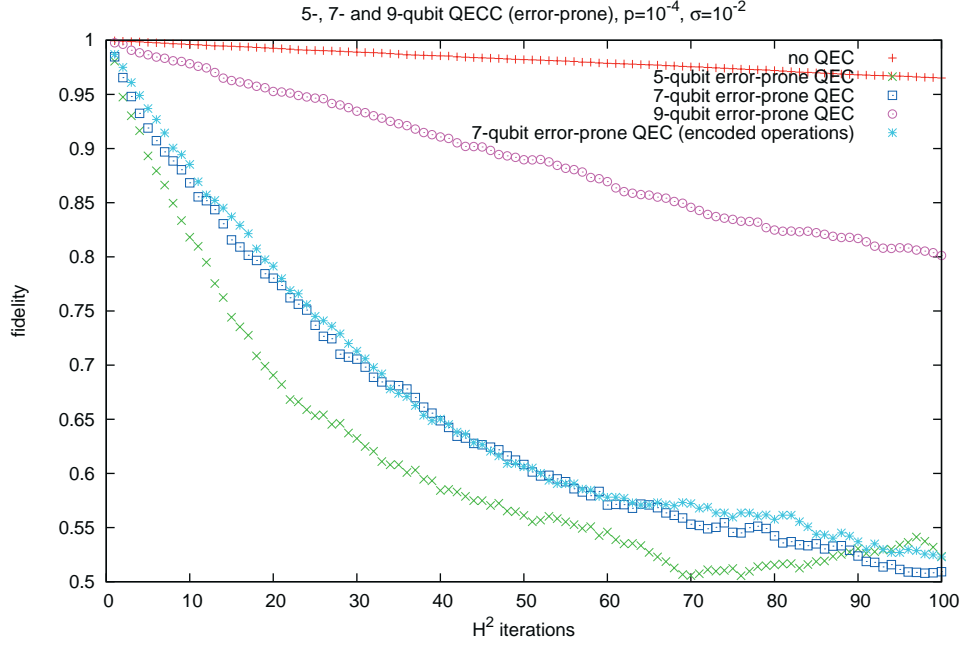
Actually, the assumption that the error correction circuits are error-free is completely unrealistic, since they consist of quantum gates themselves, which are in fact just as prone to error as the circuits they are supposed to protect. The question arises, if quantum error correction can still improve the performance of quantum algorithms, when the correction circuits do also suffer from noise. It might be that the correction circuits introduce more noise into the system than they are able to remove, so that quantum error correction is doomed to fail.



**Figure 2.45** – Example for error-free quantum error correction: 1 logical qubit is encoded into 5, 7 or 9 qubits. All codes perform equally well. They can correct any single qubit error, and a failure of two qubits within the encoded block of qubits is so unlikely in this decoherence region, that this event has not occurred during these simulations. The decrease in fidelity can be attributed to the operational imprecisions: Because we do not correct unitary over-rotations when operating on the unencoded qubit, these over-rotations accumulate over time and lead to the decrease in fidelity. In other words (see section B): The unprotected state will be susceptible to noise. In this setup we do not only stabilize qubits in memory, but we want to do computations, i.e., we want to do operations on them. For doing so, the state has to be decoded, then the operation is done and the state is re-encoded. Although no explicit decoherence operation takes place between the gate operation and the reencoding, the state is not protected against unitary over-rotations. For clarification, the 7-qubit encoding, where operations on encoded states are done, is also shown in this plot (see section 2.3.3, figure 2.49). In this case the fidelity stays constantly at 1. All errors, including operational errors can be corrected.

It turns out that error-prone quantum error correction indeed makes things even worse if applied straightforwardly (see figure 2.46). The analysis has been done again for the same parameters as in figure 2.45, the only difference being that the error correction circuits are now prone to error.

The quintessence of this analysis is that using non-fault-tolerant quantum error correction



**Figure 2.46** – *Error-prone quantum error correction.* If the correction circuits (non-fault-tolerant) are prone to errors, they introduce more noise than they can take out of the system, so the fidelity with quantum error correction is worse than the fidelity when leaving the qubit unprotected. The simulation results shown here were generated with the same parameter set as the results shown in figure 2.45. The only difference is that the error correction circuits are error-prone in this case. The 9-qubit code shows a better performance than the other codes, because the correction circuit is much shorter (see figure 2.39), thus introducing less errors. The key message is: Going to fault-tolerant quantum error correction is mandatory.

is futile, because it will introduce more errors than it can correct. Therefore, the only reasonable approach will be the extension of quantum error correction codes to fault-tolerant methods (section 2.3.3).

### 2.3.3 Fault-Tolerant Quantum Error Correction

In section 2.3.2 we have shown, that quantum error correction requires fault-tolerant methods, to be useful. That means, the quantum error correction process has to be able to remove more noise from the system than it generates, although the process is itself imperfect.

Theoretically it is proven that arbitrarily long quantum computations can be done reliably if the error rate is below a certain threshold and if there are abundant qubits available to concatenate error correction schemes. Nevertheless, today's quantum computing devices and those of the near future are and will be very limited in the number of available qubits as well as in the achievable minimum error rates. Thus, the question arises, if quantum error correction is possible, not only in theory, but also in practice, and what the thresholds for fault-tolerant quantum error correction are.

Our analyses are made for several different setups of quantum error correction. Active and passive stabilization of qubits have been examined as well as quantum operations on encoded qubits, which are necessary if not only error-protected quantum memory, but also fault-tolerant computation is required.

Our results show clear evidence that one can benefit from quantum error correction if done in a proper way, i.e. fault-tolerantly, even on limited resources, as long as the error rates are below a certain threshold. We can also make statements on the required accuracies and resources individually for specific quantum algorithms.

#### 2.3.3.1 Theoretical Principles

We describe the basic ideas of fault-tolerant quantum error correction for  $[[n,1,3]]$  codes, that encode only one single qubit into a block of  $n$  qubits. This block is then protected against a single qubit error within this block. If two errors happen in a given block, that block may fail and a decoding may result in a wrong quantum state. However, if we assume a small single qubit error rate of the order  $\mathcal{O}(p)$ , the probability of two simultaneous errors will be of the order  $\mathcal{O}(p^2)$ . In other words, a fault-tolerant quantum error correction code can convert a small physical error rate into an even smaller logical error rate.

The results of section 2.3.2 show that a non-fault-tolerant quantum error correction code by itself is only useful under the unrealistic assumption that the gates for the correction perform perfectly. If this is not the case, and this holds for all but the smallest quantum computations, a QECC will introduce more errors than it is able to correct.

Therefore, we need to encode the information according to fault-tolerant protocols. The main goal of a fault-tolerant protocol is to keep the propagation of errors under control. Obviously, a faulty single qubit gate can cause an error in the qubit involved and an erro-



**Figure 2.47** – *Example for error-propagation: A perfect CNOT gate can propagate errors to both qubits and thus can lead to a spreading of errors.*

neous two-qubit gate can induce errors in at most two qubits.<sup>20</sup> However, and this is even more important, a perfect two-qubit gate can propagate a pre-existing error to both qubits and thus leading to a spreading of errors. Figure 2.47 shows an example for the propagation of errors. A pre-existing bit flip error on the control qubit of a CNOT gate leads to a false flipping of the target qubit, so the error propagates through the CNOT and the final state is equal to a perfect CNOT followed by both a bit flip error on the control and the target qubit. Phase-flip errors on the target qubit are propagated to the control qubit, as we can immediately verify by multiplying<sup>21</sup>

$$\begin{aligned}
 \text{CNOT}(0, 1)(I \otimes Z) &= \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix} \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & -1 \end{pmatrix} \\
 &= \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & 0 & -1 \\ 0 & 0 & 1 & 0 \end{pmatrix} \\
 &= \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix} = (Z \otimes Z)\text{CNOT}(0, 1). \quad (2.120)
 \end{aligned}$$

The third kind of error, a combined bit and phase flip,  $Y = XZ$ , propagates through a CNOT gate as

$$\begin{aligned}
 \text{CNOT}(0, 1)(Y \otimes I) &= \text{CNOT}(0, 1)(X \otimes I)(Z \otimes I) \\
 &= (X \otimes X)\text{CNOT}(0, 1)(Z \otimes I) \\
 &= (X \otimes X)(Z \otimes I)\text{CNOT}(0, 1) \\
 &= (Y \otimes X)\text{CNOT}(0, 1), \quad (2.121)
 \end{aligned}$$

<sup>20</sup>This depends of course on the quantum computation device architecture. For example, this does not hold for ion trap quantum computation, where a two qubit gate is realized via a common vibration of all qubit ions (see chapter 3).

<sup>21</sup>Remember that quantum circuits are executed from left to right, while matrix multiplication is done from right to left.

and

$$\text{CNOT}(0, 1)(I \otimes Y) = (Z \otimes Y)\text{CNOT}(0, 1). \quad (2.122)$$

From the commutator relations of the Pauli and Clifford group<sup>22</sup> elements the error propagation for other types of errors and gates can be determined exactly.<sup>23</sup>

**Concatenation and the threshold theorem** As long as only one single error occurs within an encoded block, distance-3 codes, such as the 7-qubit code, can successfully correct those errors. If more than one error occurs, either directly or by propagation from another block, the block will fail. If we assume a small single qubit error rate of  $p$  for a gate or a timestep, the probability of two qubit failures in two qubits is  $p^2$ . By using a fault-tolerant protocol we can assure that a failure of a block is of the order  $\mathcal{O}(p^2)$ . The coefficient  $c$  in  $cp^2$  will depend on the code and the fault-tolerant methods, but it will be independent of the length of the overall computation. It connects the single qubit failure probability with the block failure probability. Apparently, the logical qubit gives an improvement over the single qubit if  $p < 1/c$ . In the other case, the additional qubits and gates for the quantum error correction code introduce more errors than they can correct. The improvement from  $p$  to  $cp^2$  can be driven further by going to *concatenated codes* (figure 2.48). This means that a logical bit is not only encoded in  $n$  physical bits, but in  $n$  logical bits, which are themselves encoded using  $n$  physical bits, so that in overall the logical bit is encoded using  $n^2$  qubits. This would lead to a suppression of the error probability to  $c(cp^2)^2 = c^3p^4$ . This can be generalized to  $k$  levels of concatenation, leading to an effective error probability for a logical qubit of

$$p_k = c^{2^k-1} p^{2^k}, \quad (2.123)$$

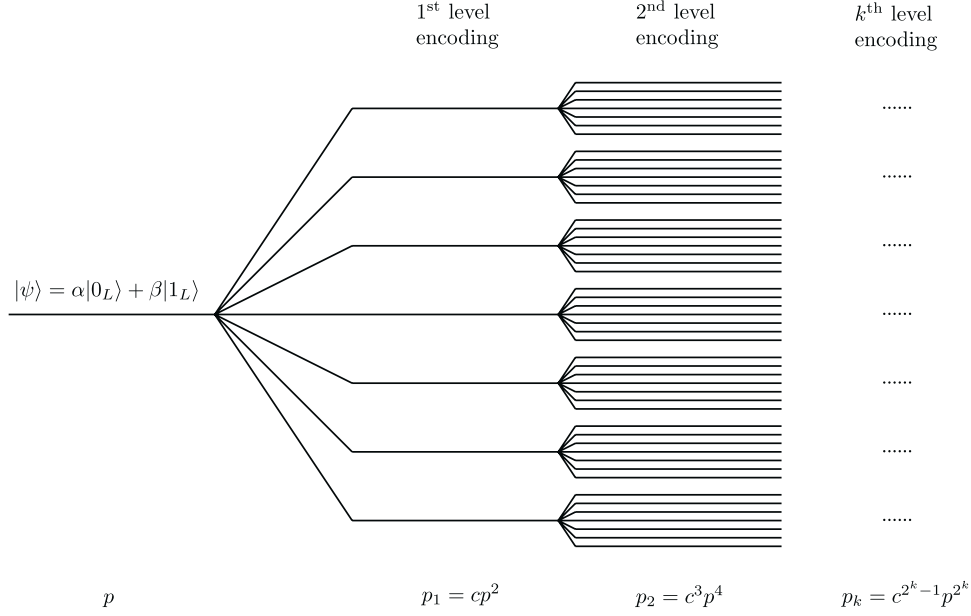
or in terms of the threshold  $p_0$ ,

$$p_k = p_0 \left( \frac{p}{p_0} \right)^{2^k}, \quad (2.124)$$

where  $p_0 = 1/c$  is called the *threshold* for fault-tolerant quantum error correction. In principle, the logical error probability can be made arbitrarily small if the single qubit error probability is below the threshold, i.e. if  $p < p_0$ . A major problem is that the number of qubits needed for multiple levels of concatenation grows exponentially with the number of concatenation levels. Nevertheless, if we want to achieve an effective error rate  $\epsilon$ , from equation (2.123) it follows that we need  $\mathcal{O}(\log(\log(1/\epsilon)))$  levels of concatenation. In overall, we need  $\mathcal{O}(\text{poly}(\log(1/\epsilon)))$  extra qubits to achieve an effective error rate of  $\epsilon$ . The resources for achieving an arbitrarily small error rate grow only polylogarithmically. The question what the precise value of  $p_0$  actually is, is answered by numerical simulations in this work in section 2.3.3.2.

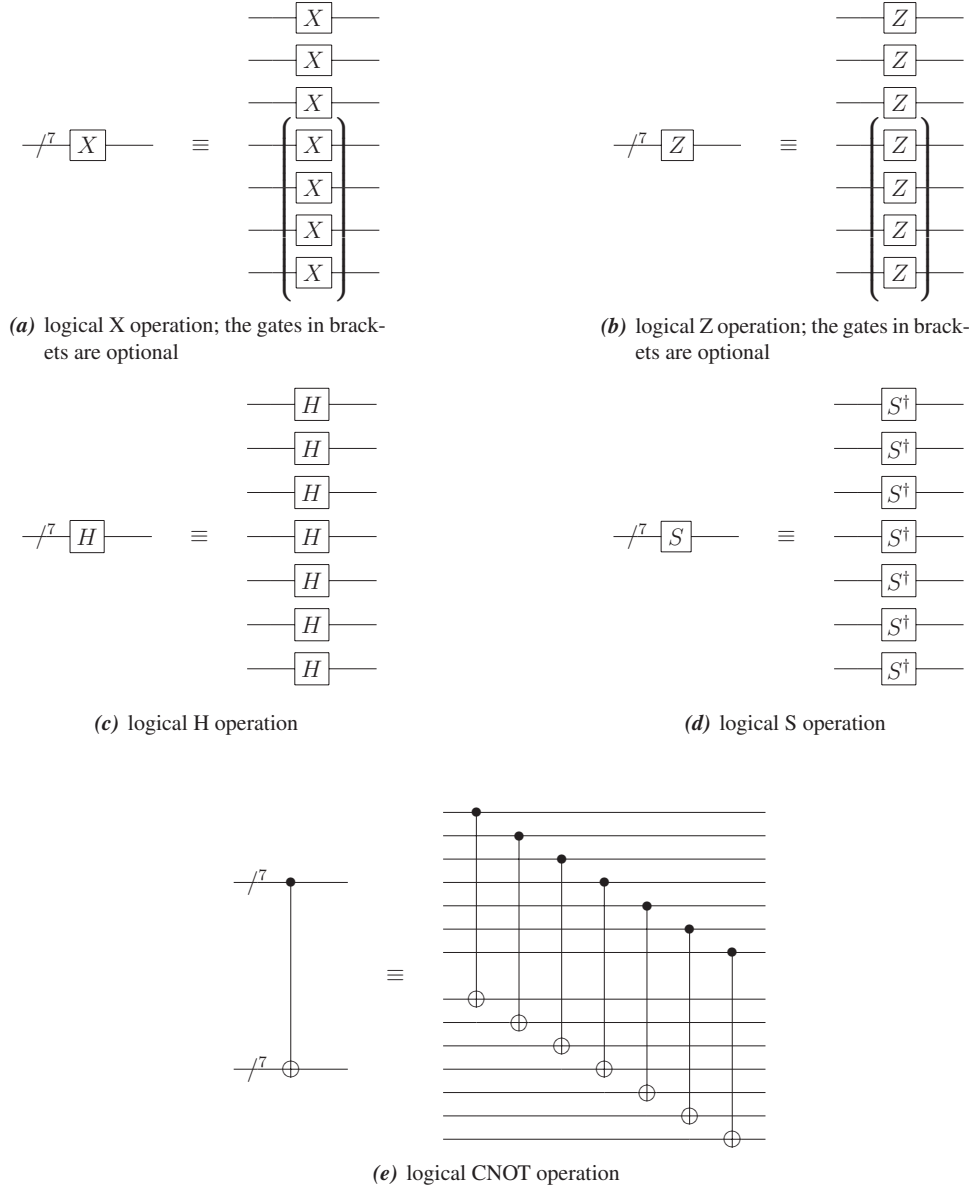
<sup>22</sup>See section C for a definition of the Pauli and Clifford group.

<sup>23</sup>Global phases can be neglected.



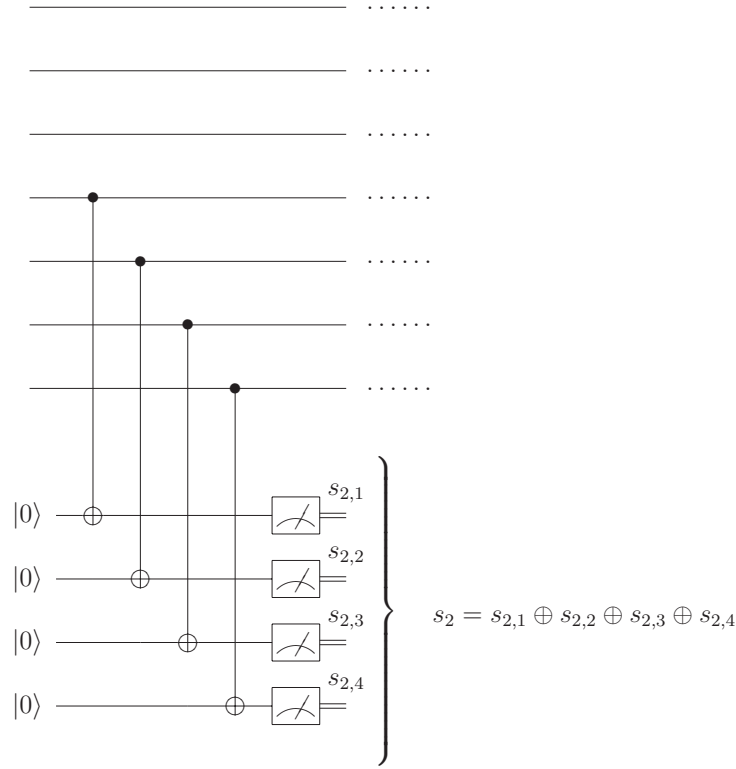
**Figure 2.48** – *Concatenated codes: A logical qubit is fault-tolerantly encoded using multiple qubits, which can be encoded logical qubits themselves. When the unencoded qubit error probability is  $p$ , using  $k$  levels of concatenation brings down the logical qubit error probability to  $p_k = c^{2^k-1}p^{2^k}$ .*

**Operations on encoded states using transversal gates** For the protection of quantum memory, i.e. fault-tolerant storing of quantum information, all quantum error correction codes considered are well suited if applied fault-tolerantly. Thus, using the 5-qubit code [Laflamme et al., 1996] may be advisable, since it uses the smallest number of additional qubits, but when going to fault-tolerant *computation*, we also need fault-tolerant implementations of a universal set of quantum gates. The necessity for doing operations on encoded logical states arises in fault-tolerant computation, because a decoding with subsequent operation and re-encoding would not only temporarily remove the protection, but it will even introduce more errors. Steane’s 7-qubit code, as described in section 2.3.1 and appendix C, is especially well suited for fault-tolerant quantum computation, because there are simple transversal gates for the Pauli and Clifford group elements [Gottesman, 1997] (see figure 2.49). For two qubit gates, transversal means that the  $i^{\text{th}}$  qubit within an encoded block interacts only with the  $i^{\text{th}}$  qubit of another encoded block of qubits. In consequence, a single error anywhere in the system can only spread to a single error per block, which can be dealt with by doing a recovery operation. Of course, two errors within a block can cause a failure of the whole block of qubits.



**Figure 2.49** – Transversal gates using Steane’s 7-qubit code. Due to the algebraic properties of the code (see appendix C and [Gottesman, 1997]), there are simple transversal representations for the logical gates. This is a specific feature of Steane’s 7-qubit code. Such straightforward transversal gates are not possible for the 5-qubit and the 9-qubit code.

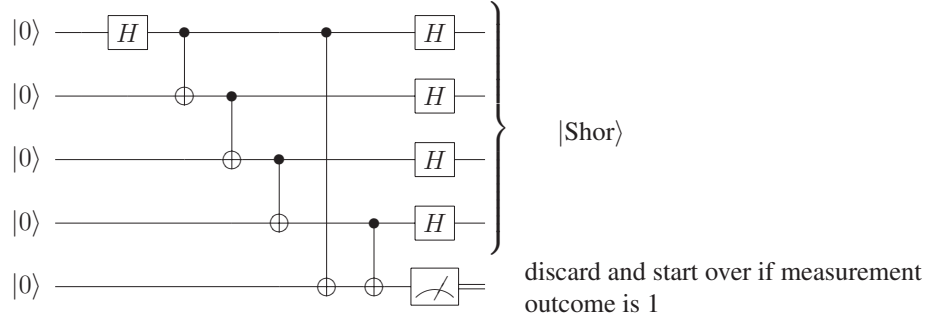




**Figure 2.50** – Part of a possible syndrome extraction circuit using four ancilla qubits instead of a single one. This procedure does not work, however, because the ancillas have been entangled not only with the error information but also with the encoded data. The measurement of the ancilla qubits gives information about the data itself and destroys the superposition of the basis states (equations (2.113) and (2.114)). The circuit shown actually measures the last four qubits of the block. If we obtain, e.g.  $|0000\rangle$  for the ancilla qubits, we have projected  $|0_L\rangle$  to  $|0000000\rangle$  and  $|1_L\rangle$  to  $|1110000\rangle$ .

**Fault-tolerant recovery methods (using Steane’s 7-qubit code)** Fault-tolerant recovery is essential for the success of quantum error correction codes. Syndrome extraction as described in section 2.3.1 is not fault-tolerant. Apparently, an error on one of the ancilla qubits used for the syndrome extraction can propagate to many qubits of the encoded block of information (see figures 2.40 and 2.47(b)). Therefore, more sophisticated methods for syndrome extraction are needed [Shor, 1996; Steane, 1997].

The idea is to use multiple ancilla qubits to extract the syndrome, so that each ancilla qubit interacts only once with one of the data qubits. This restricts the propagation of errors from a faulty ancilla qubit to multiple data qubits. The parity of the measured ancilla qubits gives the syndrome bit (figure 2.50). Nevertheless, the straightforward approach, as shown



**Figure 2.51** – Creation of the Shor state for the fault-tolerant syndrome extraction. The first Hadamard gate and the three subsequent CNOT gates create the cat state  $2^{-1/2}(|0000\rangle + |1111\rangle)$ , the Hadamard gates rotate the cat state into the Shor state (equation (2.125)).

in figure 2.50, is not feasible, because the ancillas have been entangled with the data in such a way, that a measurement of the ancilla qubits does also give information about the data qubits themselves and thereby destroys the superposition of the logical codewords  $|0_L\rangle$  and  $|1_L\rangle$ . Let us consider the example, where the four ancillas are  $|0000\rangle$  after measurement. It follows that  $|0_L\rangle$ , which has been in the superposition of all even weight Hamming codewords, is projected to  $|000000\rangle$ , and  $|1_L\rangle$  is projected to  $|111000\rangle$ .

Therefore using multiple ancilla qubits is necessary, but not sufficient for fault-tolerant syndrome extraction. Additionally, we have to use appropriate ancilla states, that do not destroy the coherence of the data. One possible way of doing this is to use an equal superposition of all even weight strings as proposed by [Shor, 1996]:

$$\begin{aligned} |\text{Shor}\rangle &= \frac{1}{\sqrt{8}} \sum_{\text{even } v} |v\rangle \\ &= |0000\rangle + |0011\rangle + |0101\rangle + |0110\rangle \\ &\quad + |1001\rangle + |1010\rangle + |1100\rangle + |1111\rangle. \end{aligned} \quad (2.125)$$

If the corresponding syndrome bit we are measuring is zero, an additional even weight string is added to the Shor state, leaving it unchanged. If the syndrome bit is one, the Shor state is transformed into the equal superposition of all odd weight strings. The parity of the ancilla qubit measurement gives the error syndrome bit, but this time we do not learn anything about the data itself while extracting the error information.

An additional obstacle to overcome is the potentially erroneous creation of the Shor state, which is depicted in figure 2.51. The circuit would not be fully fault-tolerant if only four qubits were used. A single error during the creation of the Shor state can propagate to two qubits of the Shor state and in case of two phase flip errors, they can feed back into the data block through back propagation. That is why an additional verification with the help of a

fifth ancilla qubit is necessary, which tests for multiple phase errors. Using the rules of error propagation it is easy to check, that the only way that a single error in the circuit can lead to two phase errors in the final Shor state is a bit flip error that happens between the second and third CNOT gate. Instead of getting a correct so called cat state,  $|\text{cat}\rangle = 2^{-1/2}(|0000\rangle + |1111\rangle)$ , the state before the final Hadamard rotations,  $2^{-1/2}(|0011\rangle + |1100\rangle)$ , has suffered from two bit flip errors, that result in two phase flip errors after the Hadamard operations.<sup>24</sup> Instead of getting a Shor state (equation (2.125)) we end up with the state

$$\begin{aligned} |\text{Shor}_{\text{error}}\rangle = & |0000\rangle + |0011\rangle - |0101\rangle - |0110\rangle \\ & - |1001\rangle - |1010\rangle + |1100\rangle + |1111\rangle, \end{aligned} \quad (2.126)$$

which has suffered from two phase flip errors.<sup>25</sup> To catch these multiple bit flip errors (phase flip errors) in the cat state (Shor state), it is sufficient to verify that the first and last bit of the cat state do agree.<sup>26</sup> This is done with the help of the fifth qubit and the fourth and fifth CNOT gates. Altogether, this approach ensures that the probability for two phase errors in the Shor state, which can damage the data, is of the order  $\mathcal{O}(p^2)$ .

We have not yet considered bit flip errors in the Shor state, but those are not critical, because, while they indeed lead to a faulty syndrome measurement, they cannot feed back to damage the data. In the worst case a wrong extraction of the syndrome will lead to a faulty correction step causing further damage instead of correcting the error, so additionally, the syndrome measurement must be performed multiple times to ensure the correctness of the syndrome determination up to the order  $\mathcal{O}(p^2)$ .<sup>27</sup> A (non-trivial) syndrome is considered correct (probability of failure of order  $\mathcal{O}(p^2)$ ) if it is measured twice in a row. A trivial syndrome may be mistakenly accepted, but since it requires no recovery action, there is no danger of erroneously introducing two errors into the block and the error can be reliably detected in a future round of error correction. The (non-fault-tolerant) syndrome extraction depicted in figure 2.40 is extended to a fault-tolerant version (figure 2.52) by using Shor state ancilla qubits instead of single ones.

All precautions and measures described above will ensure that the recovery can only fail if two independent errors occur, but the probability for that is of order  $\mathcal{O}(p^2)$ .

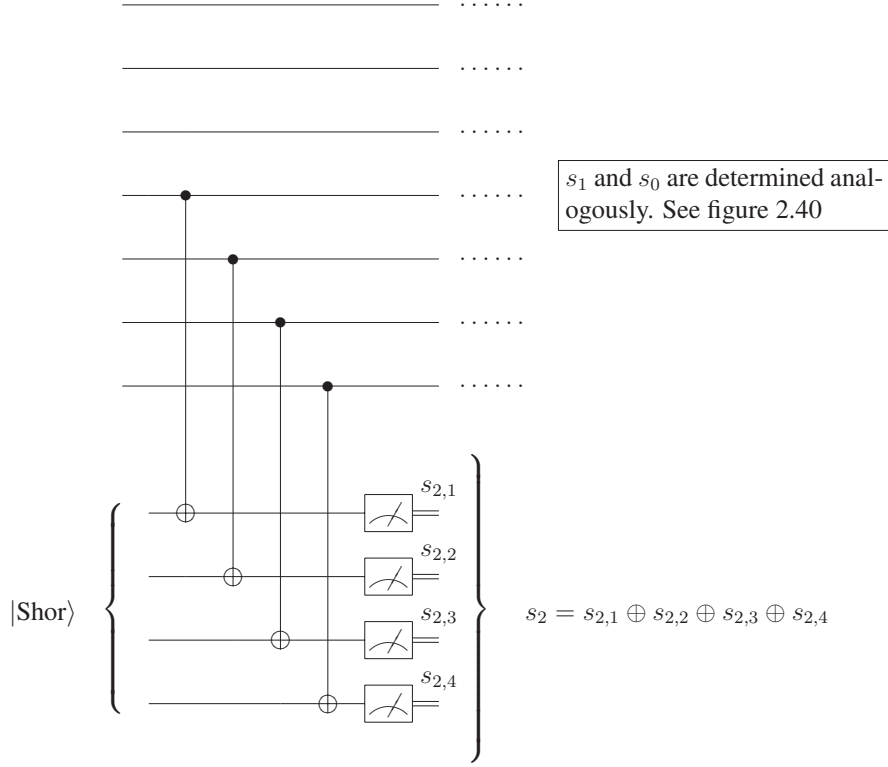
The extraction of the phase flip syndrome can be simplified [Preskill, 1998] to a less complex circuit by using the identity in figure 2.53. The creation of the Shor state involves a

<sup>24</sup>The symmetry of the Shor state simplifies the analysis, because the cases of the cat state with one or three bit flip errors all lead to a Shor state with effectively a single phase flip error and a cat state with all four bits flipped does not change at all.

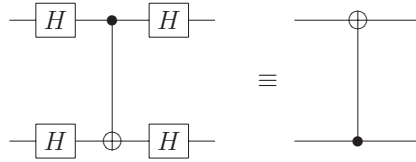
<sup>25</sup>Again, symmetry does not tell if these are qubits 0 and 1 or qubits 2 and 3.

<sup>26</sup>This will additionally catch the one and three error cases.

<sup>27</sup>Keep in mind that during the ancilla preparation, ancilla verification, syndrome extraction and syndrome verification the data qubits will suffer from decoherence. Nevertheless, if done correctly, i.e. fault-tolerantly, we can still ensure the improvement of the block error rate to  $\mathcal{O}(p^2)$  compared to the single qubit error rate of  $\mathcal{O}(p)$ , provided that we are below the threshold. This indeed becomes smaller as the complexity of the recovery circuit increases.



**Figure 2.52** – Fault-tolerant syndrome measurement circuit for Steane’s 7-qubit code. The creation and verification of the Shor state is depicted in figure 2.51. The syndrome bits  $s_i$  are given by the parity of the four ancilla qubits each. Shown here is the determination of  $s_2$ . The syndrome bits  $s_1$  and  $s_0$  are extracted analogously to the scheme in figure 2.40 using verified Shor states instead of a single ancilla qubit. Note that the syndrome extraction is possibly repeated multiple times.



**Figure 2.53** – Hadamard rotations rotate the basis, such that the control and target qubit of a CNOT are interchanged.



**Figure 2.54** – The extraction of a single bit of the phase flip syndrome is depicted schematically. The Hadamard gates are transversal gates operating on 7 and 4 qubits. The CNOT consists of four basic CNOT operations according to figure 2.40. The simplified circuit saves the transformation of the data qubits into the Hadamard rotated basis.

creation of a cat state with subsequent rotation into the Shor state (figure 2.51). For the phase flip syndrome measurement the identity shown in figure 2.54 can be exploited.

**Fault-tolerant encoding/decoding methods (using Steane’s 7-qubit code)** We are able to do a fault-tolerant recovery, but the question how to do a fault-tolerant encoding/decoding still remains. If the task would be to create a known encoded quantum state, such as  $|0_L\rangle$  or  $|1_L\rangle$ , a recovery operation and verified measurement will project the state into one of the desired states (with optional bit flip of the logical state). But if we are forced to encode an unknown quantum state, we must use the encoding circuit (figure 2.42), and since no measurement can verify the encoding, the fidelity of the encoded state will inevitably be  $F = 1 - \mathcal{O}(p)$ . This is the reason, why quantum error correction can only be efficient for algorithms which are not too short. An encoded memory qubit that suffers a fidelity loss due to the encoding can still be protected for a long time in contrast to the unencoded qubit. With the help of encoded operations fault-tolerant operations can be done while the qubit remains protected, which would not be possible for an unencoded qubit. The encoding step will unavoidably be associated with a fidelity loss of  $\mathcal{O}(p)$ , but nevertheless, an algorithm can benefit from keeping the qubits encoded and protected during its execution. If we want to measure a qubit at the end of an algorithm, this can be done by decoding (with  $F = 1 - \mathcal{O}(p)$ ) and subsequent measuring or with a parity measurement to decide if the state is  $|0_L\rangle$  or  $|1_L\rangle$ . Since a single error can lead to a faulty parity measurement, this procedure has to be repeated twice to ensure a failure probability of  $\mathcal{O}(p^2)$ .

### 2.3.3.2 Numerical Simulations

With our universal quantum computer simulator at hand, which has been extended to quantum error correction capabilities and especially fault-tolerant methods, we are able to analyze the performance of quantum error correction. Our analyses showed that only fault-tolerant approaches can give any advantage over non-error corrected circuits, so here we concentrate our studies on Steane's 7-qubit quantum error correction code, where we have a collection of fault-tolerantly implemented gates. We focus on algorithms where only gates of the Clifford group are used and leave the very complex fault-tolerant implementation of the T gate and the Toffoli gate for future work.<sup>28</sup> The basic ideas of fault-tolerant quantum computation and threshold determination were formulated in section 2.3.3.1.

Threshold determinations from numerical simulations have been done by several people [Zalka, 1996; Steane, 2003; Aliferis et al., 2005; Reichardt, 2006], but all of these simulations were based on tracking the error propagation instead of following the evolution of the state vector. This leads to many oversimplifications, e.g. missing the relevance of the ancilla qubits or other unrealistic assumptions. More importantly, it is not easily possible in such an approach to deal with operator imprecision and systematic errors in such a straightforward way as our simulation can do. Our simulation is based on a well defined constructive procedure (see section 2.3.3), that also allows for the adoption to a variety of experimental setups.

**Assumptions made for the simulations** We need to emphasize the physically reasonable assumptions made for the simulations, since any statement about thresholds depends substantially on these assumptions. We have chosen an error model with statistically independent random errors that are assumed to be uncorrelated. Depending on what physical realization is chosen, it might be possible that nearby qubits are more likely affected by correlated errors. This is not taken into consideration in our general error model. Furthermore, we define our error rate independent of the system size.<sup>29</sup>

We also will assume maximal parallelism, i.e., we can execute many gates in parallel in a single timestep. This is essential in the determination of our threshold. Without parallelism the fault-tolerant threshold condition becomes practically impossible to achieve, because errors accumulate faster than error correction can correct them. In ion trap quantum computers for example, parallelism is hard to achieve, because all qubits couple via a common phonon bus (see section 3.1). A possible exploitation of different phonon modes could at least provide some parallelism, however.

Another assumption we made is that we can perform two-qubit gates on any distant pair of qubits. For some architectures that use nearest neighbor interactions, this assumption does

---

<sup>28</sup>A proposal for a fault-tolerant implementation of the T gate can be found in [Fowler, 2005]. For a fault-tolerant implementation of the Toffoli gate see for example [Preskill, 1998].

<sup>29</sup>Depending on specific hardware architectures, e.g. for ions in a single trap, the error rate will exhibit a system size dependency.

not hold. However, a promising future quantum computer architecture should allow for arbitrary two-qubit operations. Both issues mentioned can be considered as a requirement for future quantum computing architectures.

We do not make the unrealistic assumption that we have a large or even unlimited resource of fresh ancilla qubits available. So the determination of the syndrome bits (figure 2.40) is done sequentially and not in parallel. Since six syndrome bits have to be determined for one correction step, parallel extraction would require 30 qubits just for the ancillas. We extract the syndrome bits one after the other and reuse our reset ancilla block each time.

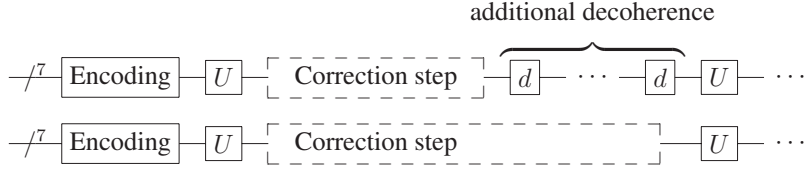
Additionally, we assume that we can reset our ancillas in a single timestep. From a thermodynamic point of view our ancillas carry entropy out of the system, so at some point they have to be reset with a dissipative operation, so they can be reused. This reset operation requires a cooling of any device and might take several timesteps depending on the device architecture.

A similar question can be asked for a measurement process. Here we assume that a measurement takes exactly one timestep. For slow measurements ( $\approx 100$  timesteps) the threshold may be one order of magnitude lower [Steane, 2003].

By using the depolarizing channel as our decoherence model we assume that bit flip and phase flip errors occur with equal probability. If for a device in the lab these probabilities would be non-equally distributed, one could tailor the correction schemes to the error model and achieve a higher threshold.

Overall, we can say that in this analysis we choose a rather general approach, not referring to a specific device architecture yet. Nevertheless, the simulation code can be easily adjusted to physical device specifications.

**Threshold Determination** For the threshold determination the question is asked, under which conditions a block of encoded qubits performs better than a single unencoded qubit. We answer this question by examining the  $H^{2k}$ -algorithm for a single qubit encoded with Steane's 7-qubit code into a block of seven qubits. For the syndrome extraction we decided to use Shor state ancillas [Shor, 1996], because we are limited by the number of qubits that we can simulate on our computers (just as experimentalists are limited by the number of realizable qubits as well), and the Shor state offers a way to extract the syndrome fault-tolerantly by using the minimum number of additional ancilla qubits. A Shor state consists of 4 ancilla qubits, and one additional qubit is needed for the verification of the Shor state (see figure 2.51 and equation (2.125)). A logical data qubit consists of 7 single qubits. An encoded qubit with corresponding ancillas requires  $7 + 5 = 12$  physical qubits. In total, a fully encoded system of  $n_L$  logical qubits would require  $n = (7 + 5) * n_L$  qubits. With the current status of the JUGENE system we are limited to  $n = 40$  qubits. In order to allow the simulation of larger systems, we reuse the ancilla memory space in our simulations and assume that, in practice, the ancilla operations could be done in parallel on different logical



**Figure 2.55** – Decoherence on idling qubits. Example with two logical qubits.  $U$  denotes a unitary logical gate operation and  $d$  indicates a decoherence operation. The correction step does not always have a fixed length, because sometimes the syndrome extraction has to be repeated or the ancilla preparation may fail and requires a repetition. Our simulation is realistic in the sense that it accounts for idling qubits, that are affected by decoherence while idling.

qubits. This is not a real sharing of ancilla qubits with each logical qubit, since this would make parallel operations impossible, but it is rather a simulational dodge that saves memory space, so that a simulation of  $n_L$  logical qubits requires an effective memory storage of only  $n = 7n_L + 5$  qubits.

The operations on different logical qubits are in fact done sequentially in the simulation, but the timesteps are adjusted in such a way, as if the operations could be done in parallel. If operations on one of the logical qubits take more time than on one of the others this is realistically accounted for: All qubits are synchronized, so the idling qubits do suffer from decoherence during their waiting time (see figure 2.55).

To ensure the correctness of the syndrome measurement we decided to run the syndrome extraction loop until we measure the same syndrome twice in a row or until we measure the syndrome 0. This makes sure that the probability of introducing two errors within a block is of the order  $\mathcal{O}(p^2)$ . In case the extraction of the syndrome 0 is faulty, we will not introduce a second error by trying to correct for the wrong error, but the error is propagated further and can be reliably detected in a future error correction step.

As described in section 2.2.1 our simulations are based on statistically independent error locations. We have to make sure to gather sufficiently high statistics. “Sufficiently high” depends on the single qubit error probability  $p$ , the length of the algorithm, the number of qubits, the complexity of the algorithm, i.e. the phase space of the system, and of course on the required accuracy of the result.

As an example, running the  $H^{2k}$ -algorithm with the parameter set  $k = 1000$ ,  $p = 10^{-7}$  and fully fault-tolerant Steane encoding requires at least  $m = 10000$  statistical iterations to yield a result with a relative error<sup>30</sup> of less than  $10^{-3}$  in the final fidelity.<sup>31</sup> Running this system on a single processor of the JUMP (Power4) system requires about two days of computation

<sup>30</sup>We take the standard deviation of the mean as a measure of statistical error.

<sup>31</sup>Note that we use the fidelity here as a measure of correctness, because error propagation might lead to global phase factors. For example the Hadamard gate anti-commutes with the Pauli operators,  $\{H, \sigma_i\} = 0$ , i.e., if an error is propagated through a Hadamard gate, we gather a global phase factor of -1.



time for  $m = 10000$  iterations. We will see that for an exact direct determination of the threshold an even higher accuracy and therefore more statistical iterations are required.

**Reset operation** For the syndrome extraction additional ancilla qubits are used. As we do not make the unrealistic assumption that we have an unlimited resource of fresh ancillas available, we have to reuse our ancillas. This is done by resetting them to their original state. A reset operation is a trivial task in classical computation, whereas in quantum computation this has to be handled carefully. A reset operation carries away entropy out of the system and is a non-unitary evolution of the system that cannot be handled in the usual manner. For our simulations we assume that a reset operation requires the same time as a gate operation. The reset operation is done in several steps. First, a projective measurement is done, clearing the amplitudes of the state vector, which do not belong to the measurement outcome. If the measurement outcome was 1, a flip of the ancilla qubit to the state 0 is necessary. Finally, since the clearing of amplitudes is a non-unitary operation, a renormalization of the state vector is necessary.

**Rough estimation to determine interesting range of parameters** We will concentrate on Steane's 7-qubit code using fault-tolerant methods, i.e. with  $4 + 1$  ancilla qubits, and we will always use encoded operations. Using real fault-tolerant methods, the frequency of error correction steps can be chosen to be maximal, which grants the best correction performance. Obviously the best results are gained when decoding is completely avoided during intermediate steps of the algorithm.

So the main two error parameters that have to be examined are the decoherence probability  $p$  and the measure for the operational error imprecision  $\sigma$ . The maximum value of  $\sigma$  has been set to  $\sigma = 0.1 \hat{=} 5.7^\circ$ , which is rather large, because most experimental setups can perform better than this. The minimum value for sigma is set to  $\sigma = 10^{-3}$ , because for smaller values the difference to  $\sigma = 0$  is negligible.

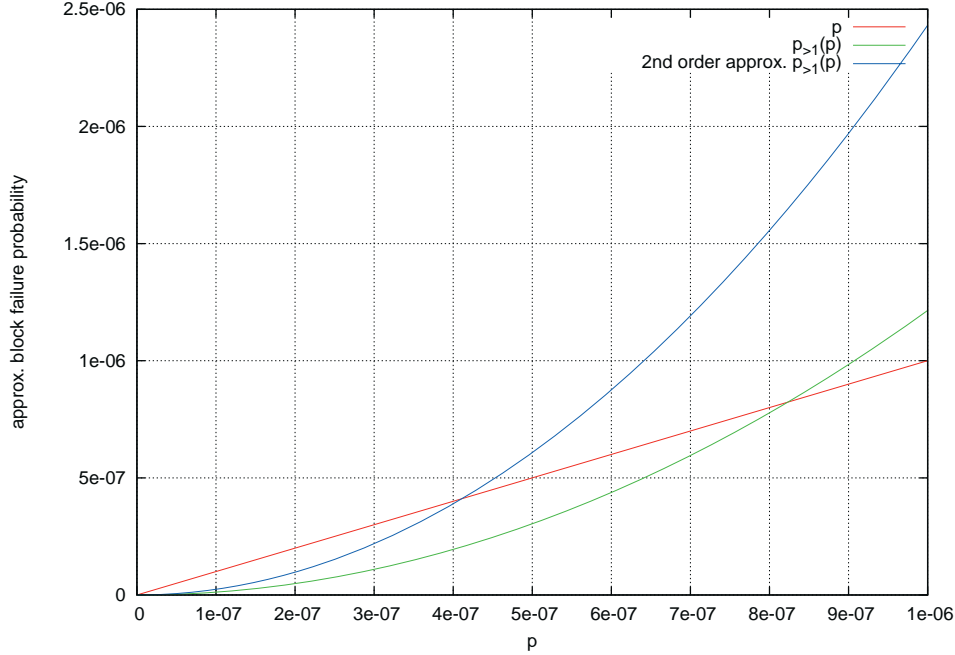
Unfortunately, the determination of the parameter range for  $p$  is not that easy. The parameter range chosen for the analyses in section 2.2 are magnitudes above the fault tolerant threshold, so we first have to make an estimate for the magnitude of the threshold.

A rough estimate of the failure probability of a block of qubits can be done: Let  $p$  be the error probability of a single qubit. If we look at a block of  $n$  qubits during  $m$  timesteps, i.e., if we look at the area size of  $n \times m$ , we can ask for the probability of more than one failure  $p_{>1}$  within this area. The probability  $p_k$  for exactly  $k$  failures is given by

$$p_k = \binom{nm}{k} p^k (1-p)^{nm-k}. \quad (2.127)$$

The probability  $p_{>1}$  is given by

$$p_{>1} = 1 - p_0 - p_1 = 1 - (1-p)^{nm} - nmp(1-p)^{nm-1}, \quad (2.128)$$



**Figure 2.56** – Approximate block failure probability depending on the single qubit error probability  $p$ . Rough estimate of the expected range where the threshold is supposed to be found, i.e., where the block failure probability becomes smaller than the single qubit error probability. Loosely speaking, a block fails if more than a single error occurs within the block.

and a second order approximation yields

$$p_{>1} \approx nm(nm - 1)p^2. \quad (2.129)$$

We can now estimate the probability of two errors occurring within this area of  $n$  qubits in  $m$  timesteps. If we assume an encoding with Steane's 7-qubit code with additional 5 ancilla qubits, i.e.  $n = 12$ , we can estimate the failure probability for the correction step circuit. This is not a fixed time operation, because neither the ancilla state preparation nor the syndrome extraction is deterministic. If we assume that the ancilla preparation into the Shor state is always successful and the syndrome measurement is always performed twice, a correction step takes about  $m = 130$  timesteps, and we get the result shown in figure 2.56. Note, that this is not a detailed analysis, but a rather rough estimate, where we haven't considered issues like:

- the preparation of the Shor state can fail,
- the syndrome measurement can be performed more or less than twice,
- we are not distinguishing between ancilla and data qubits,

- the 7-qubit code can actually correct two X or Z errors, even on different qubits,
- and errors can even cancel out each other, e.g.  $XX=I$ .

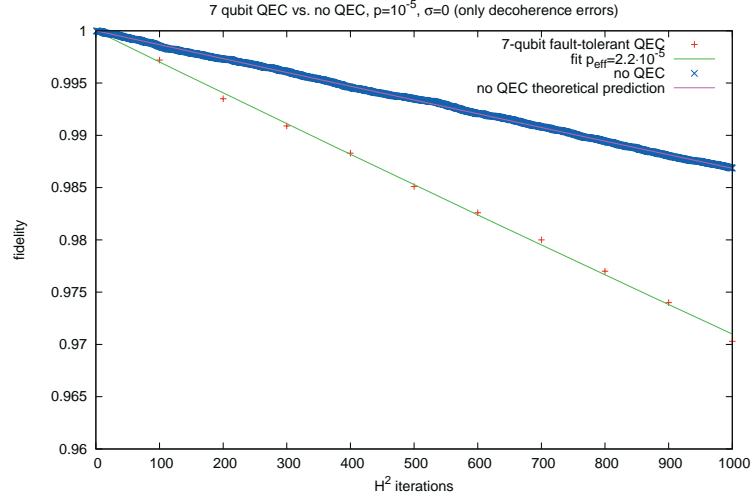
A detailed analysis of error propagation has to be done, but analytically this is not feasible. Therefore, we carry out simulations to give an answer to this problem. The rough estimate shows, that we would need an error rate of the order of  $8 \cdot 10^{-7}$  for fault-tolerant quantum error correction to perform better than using no quantum error correction at all. This is supported by simulation runs with  $p$  being in the range  $10^{-6}$  to  $10^{-2}$  in section 2.2. In this high error probability regime, it is definitely better to abandon quantum error correction. The determination of the exact threshold without neglecting too many important issues is done using numerical simulations.

**Coarse estimate of the threshold** For a coarse estimate of the threshold we ran our simulations for several parameters of  $p$  both for the non-encoded and encoded case. Figure 2.57 shows two example runs with  $p = 10^{-5}$  and  $p = 10^{-6}$ . By choosing to plot against the number of  $H^2$ -iterations we can use the smoothness of the fidelity curve as a consistency check to ensure that we have collected enough statistics. The simulations with  $p = 10^{-5}$  were done with  $m = 10000$  statistical iterations, those with  $p = 10^{-6}$  with  $m = 100000$  iterations. The drawback of this method is that we have to decode the state intermittently to measure the fidelity, but this decoding and re-encoding will have an effect on the fidelity itself. We try to minimize this effect by decoding after each 100<sup>th</sup>  $H^2$  operation only. We can be sure of the lower bound of  $p = 10^{-6}$  for the threshold, but the upper bound of  $p = 10^{-5}$  can only be an approximation. Another approach would measure the fidelity only at the end of the algorithm and use the standard deviation of the mean as a measure of the stochastic relevance. A brief check reveals that this approach is indeed too coarse for an exact determination of the threshold: If we assume a quadratic relation between the single qubit error rate and the block error rate,  $p_{\text{eff}}(p) = cp^2$  (figure 2.56), we can calculate the constant  $c$  and the threshold  $p_{\text{thr}} = 1/c$ . Doing this for both cases reveals a threshold of  $p_{\text{thr}_1} = 1.1 \cdot 10^{-6}$  for  $p = 10^{-6}$  and  $p_{\text{thr}_2} = 4.5 \cdot 10^{-6}$  for  $p = 10^{-5}$ . Therefore, determining an exact threshold requires to omit the intermediate decoding steps and to gain a higher accuracy, which means that we need even more statistical iterations. We choose another approach to determine the exact threshold, that is less prone to statistical fluctuations.

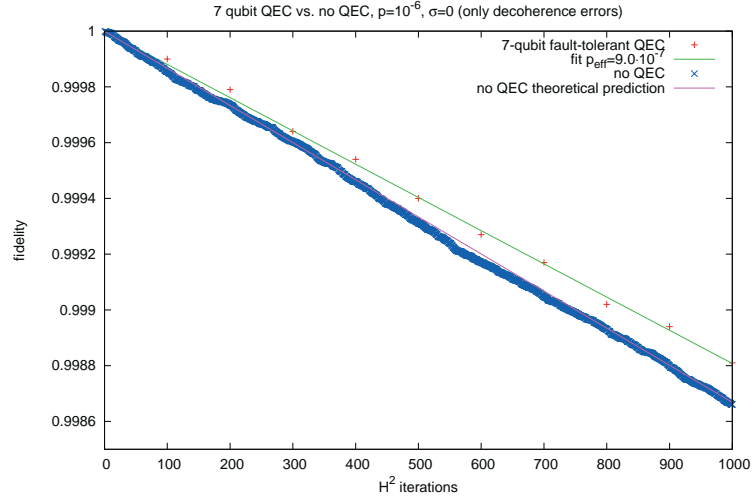
**Fine determination of the threshold** The effort to determine the exact threshold would be enormous, if done straightforwardly, e.g. with nested intervals, because locating the threshold with very high accuracy would require too many statistical iterations. Therefore, we prefer another approach: For the  $H^{2k}$ -algorithm we know, in principle, how the fidelity decreases for an unprotected qubit (equation (2.84)):

$$f(k) = \left( \frac{1 + \left(1 - \frac{4}{3}p\right)^{2k}}{2} \right)^n. \quad (2.130)$$

## 2.3. SIMULATION OF QUANTUM ERROR CORRECTION



(a) above threshold



(b) below threshold

**Figure 2.57** –  $H^{2k}$ -experiment: Comparison of the single unencoded qubit case with Steane's 7-qubit quantum error correction scheme. The fit gives an effective error rate  $p_{\text{eff}}$  for the encoded block. The plots demonstrate that the threshold is above  $p = 10^{-6}$  and probably below  $p = 10^{-5}$ . The effects of the intermittent measurements of the fidelity might lead to a decrease in fidelity. The simulations with  $p = 10^{-5}$  were done with  $m = 10000$  stochastic iterations, those with  $p = 10^{-6}$  with  $m = 100000$  iterations.

Now we have a protected qubit with an effective error probability  $p_{\text{eff}}$  that degrades like

$$\hat{f}(k) = \left( \frac{1 + \left(1 - \frac{4}{3}p_{\text{eff}}\right)^{2k}}{2} \right)^n, \quad (2.131)$$

with

$$p_{\text{eff}} = cp^2, \quad (2.132)$$

because we use a fault-tolerant design. We look at the gain  $g$  that we get by using quantum error correction, i.e., we examine the ratio

$$g(p) = \frac{\hat{f}(k)}{f(k)}(p) = \left( \frac{1 + \left(1 - \frac{4}{3}cp^2\right)^{2k}}{1 + \left(1 - \frac{4}{3}p\right)^{2k}} \right)^n. \quad (2.133)$$

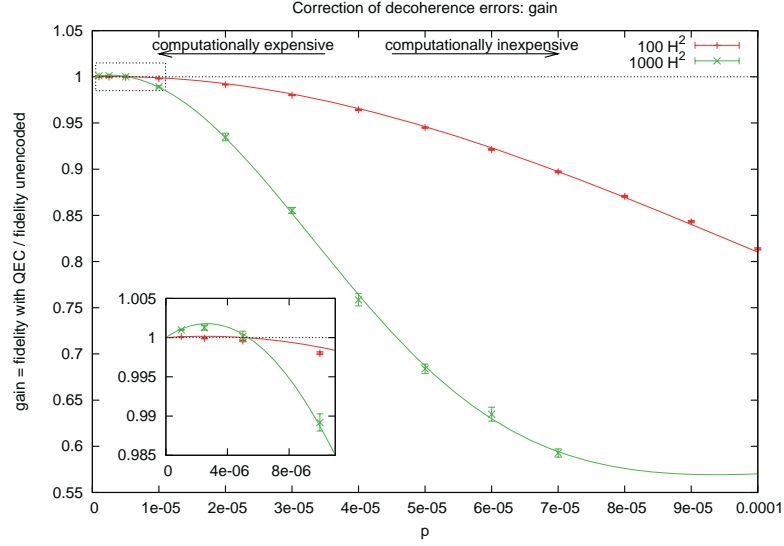
The threshold  $p_{\text{thr}}$  is given by  $g = 1$  or  $p_{\text{eff}} = p$ , i.e.

$$p_{\text{thr}} = \frac{1}{c}. \quad (2.134)$$

Note that the threshold  $p_{\text{thr}}$  is independent of the system size  $n$  and the length of the algorithm  $k$ . In contrast, the gain  $g$ , of course, depends on the length of the algorithm as well as on the system size.

Choosing this approach we can run the simulations for a wide range of error rates  $p$  and determine the constant  $c$  as well as the threshold  $p_{\text{thr}}$  from a weighted fit to the data that also accounts for the statistical uncertainties. The approach is outlined in figure 2.58. Depending on the error rate  $p$  more or less statistical iterations are necessary. For larger  $p$  the simulation becomes less expensive whereas for smaller  $p$  a high number of statistical iterations is needed and much of the simulation time goes into gathering enough statistics. Instead of running many simulations around the range where the gain becomes 1, trying to locate the threshold directly, we extend the analysis to a range of higher error rates, where the simulation cost is lower. We just have to make sure that we don't use data where the fidelity is already saturated at  $\hat{f}(p) = 0.5$  as shown in figure 2.59. If this is the case, we cannot gain information about the constant  $c$ , but the curve progression is determined by the fidelity of the unencoded case only, while the encoded case results in complete noise.

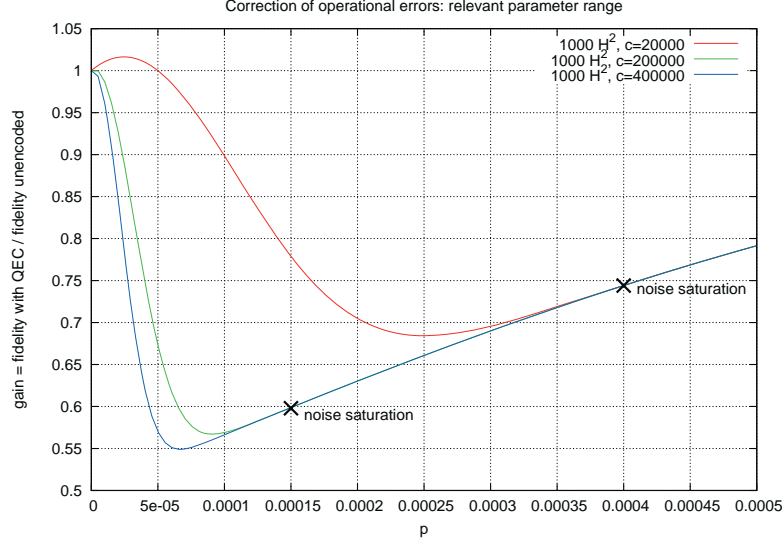
Since we use a stochastic error model, we have to do many simulation runs to determine a single data point in figures 2.58 and 2.61. The standard deviation of the mean fidelity for the statistical fluctuations are an indicator for the deviation from the true value, but since we cannot gather an infinite amount of statistics, we can only approximate the true value. Taking the standard deviation of the mean as the error might underestimate the error: Figure 2.60 shows an example of the mean fidelity and standard deviation of the mean depending on the number of stochastic iterations. As one can see the  $(i + j)^{\text{th}}$  error interval indicated by the standard deviation is not necessarily completely included in the  $i^{\text{th}}$  error interval, so stopping the iterations at the  $i^{\text{th}}$  step and taking the standard deviation at that



**Figure 2.58** – Achievable gain by using fault-tolerant quantum error correction: The ratio between the fidelity with quantum error correction and without quantum error correction is plotted against the single qubit error rate  $p$ . Two different algorithm lengths are shown. The data points are results from numerical simulations, whereas the curves are weighted least squares fits to the data (using equation (2.133)). The threshold can be determined from the intersection with the dotted line. The interesting region is framed by the dotted box, which is also magnified in the plot (small frame). A very high precision would be needed to locate the point of intersection from datapoints only. This high precision can only be obtained by accumulating very high statistics requiring excessive amounts of computing time. This can be avoided by running simulations in the higher  $p$  region, which is computationally less expensive, and doing a weighted least squares fit to determine the threshold. The threshold should be independent of the algorithm length, i.e., the intersection point with the horizontal one axis should be the same for any length of the algorithm.

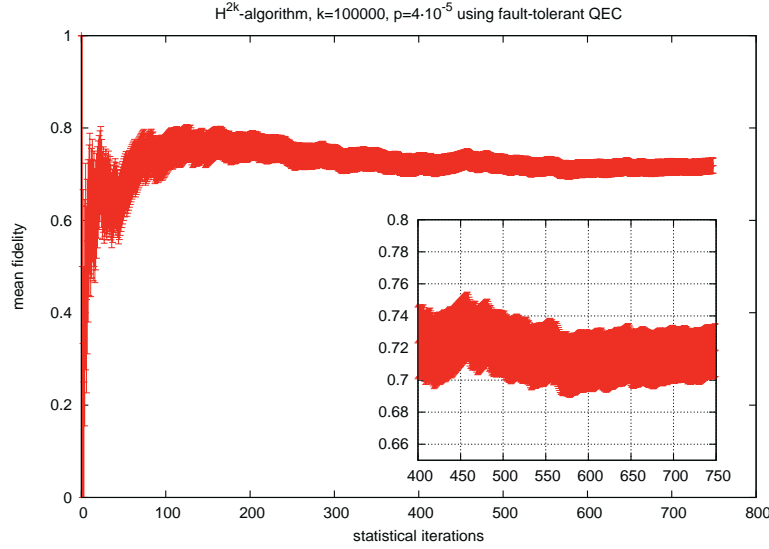
point may underestimate the error. That is why for each point in figure 2.61 we always take a look at the mean fidelity first and make sure that we gathered statistics high enough to be in an area without high fluctuations and where the mean fidelity is close to the asymptotic value.

Figure 2.61 summarizes the result of the threshold determination for decoherence errors. Here the gain is plotted against the single qubit decoherence rate  $p$ . This plot contains the results of three different lengths of the  $H^{2k}$ -algorithm,  $k \in \{100, 1000, 100000\}$ . The gain is a direct indicator for the range of parameters, where there is benefit from using quantum error correction. In this plot, the interesting part is contained on the left side of the plot, where the gain becomes larger than 1. The points on the right side, which are lower than 1,



**Figure 2.59** – Example of the gain for several parameter values of  $c$  (theoretical curves). For the determination of the constant  $c$ , only measurements in the range, where the fidelity of the encoded case has not yet reached noise saturation, can give information.

are nevertheless useful for the determination of the threshold as described before. But if  $p$  is of this size, the use of quantum error correction is useless, because a non-encoded system will always perform better. In this case, quantum error correction introduces more errors than it can correct, even if done fault-tolerantly. The lines in figure 2.61 are fits through all points of each fixed  $k H^{2k}$ -iteration run (see equation (2.133)). This is possible, because we know the theoretical decrease of fidelity for this algorithm depending on the effective error rate, for which fault-tolerant approaches guarantee  $p_{\text{eff}} = cp^2$ . The result of a weighted least squares fit gives the unknown parameter  $c$  and thus the threshold  $p_{\text{thr}} = 1/c$ . The intersection of the curves with the  $g = 1$  axis locates the threshold for fault-tolerant quantum computation. Apparently, the exact intersection point is not easy to identify from the plot, especially for the short algorithms, because the gain is marginal and barely rises above 1. Straightforward determination by nested intervals would require a very high resolution in  $y$ -direction and therefore a tremendous amount of statistical runs. By weighted fitting we eliminate the necessity for a very high resolution. Keep in mind that each point is the result of many statistical iterations and especially for low values of  $p$  the computational effort can increase dramatically to gather sufficiently high statistics. The computational effort for going to longer algorithms increases linearly for this simple algorithm. That is why the error bars for the runs with  $k = 100000$  iterations are large, because the computational costs prohibit the gathering of very high statistics.



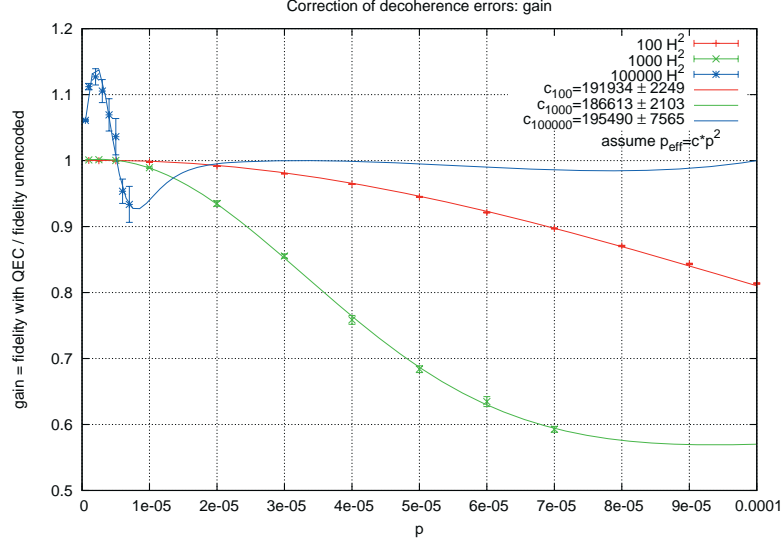
**Figure 2.60** – Mean fidelity and its standard deviation against iteration number in a fault-tolerantly protected  $H^{2k}$ -experiment with  $k = 100000$  and decoherence probability  $p = 4 \cdot 10^{-5}$ : This is an example of how to obtain the fidelity for a single data point in figures 2.58 and 2.61. Our stochastic error model requires many simulation runs to gather enough statistics. We have to make sure that the value we determine is close to the asymptotic value. The small box is a magnification of the right part of the larger plot. It shows that taking the standard deviation of the mean as the error might underestimate the error, because there are still fluctuations: For example, stopping at iteration number 450 could have underestimated the error, because the value for the mean fidelity at a higher iteration step, e.g. at iteration 580, could lie outside the range indicated by the error bars at iteration 450.

The intersection point of the curves with the  $g = 1$  axis should be independent of the algorithm length as well as the system size (equations (2.133) and (2.134)). Indeed, the curves of all three fits give almost the same result: They all intersect in one point (and of course in the trivial point (0,1)). Weighted least squares fits for three different algorithm lengths  $k$  give the fit results  $c_{100} = 191934 \pm 2249$ ,  $c_{1000} = 186613 \pm 2103$  and  $c_{100000} = 195490 \pm 7565$ , where the errors are the asymptotic standard errors of the least squares fit. In terms of thresholds those values are  $p_{\text{thr}_{100}} = (5.21 \pm 0.07) \cdot 10^{-6}$ ,  $p_{\text{thr}_{1000}} = (5.35 \pm 0.07) \cdot 10^{-6}$  and  $p_{\text{thr}_{100000}} = (5.12 \pm 0.1) \cdot 10^{-6}$ . The mean value is  $p_{\text{thr}} = (5.23 \pm 0.14) \cdot 10^{-6}$ . A conservative estimate for the threshold of fault-tolerant quantum error correction is

$$p_{\text{thr}} = (5.2 \pm 0.2) \cdot 10^{-6}. \quad (2.135)$$

If we want to give a safe lower bound for the threshold we can state with confidence that the threshold is  $5.0 \cdot 10^{-6}$ , so that for any single qubit error rate less than this value we



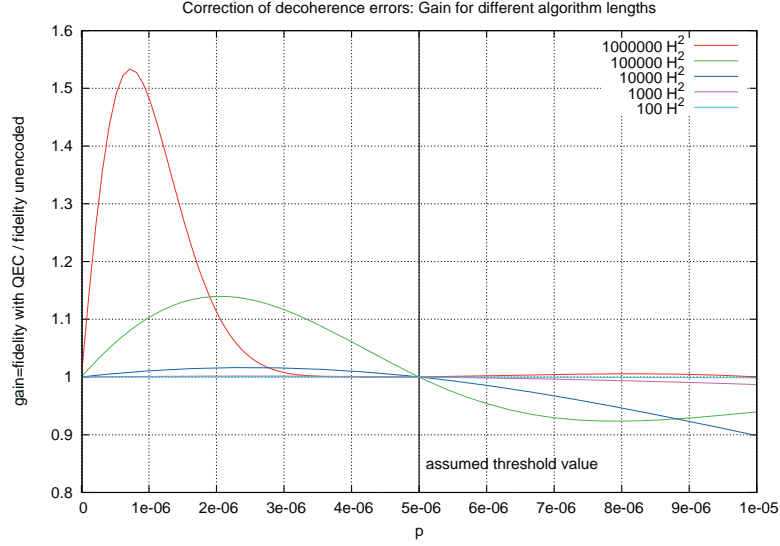


**Figure 2.61** – Achievable gain by using 7-qubit fault-tolerant quantum error correction to protect against decoherence errors. The ratio between the fidelities with and without quantum error correction is plotted against the single qubit error rate. Each point corresponds to a numerical simulation run of the  $H^{2k}$ -experiment ( $k \in \{100, 1000, 100000\}$ ) for each the quantum error correction case and the unencoded case. The curves are the results of weighted least squares fits using equation (2.133)). For a single logical qubit the fidelity reaches 0.5 in the large noise limit, so the maximum gain is 2 in this case. The threshold for fault-tolerant quantum computation is  $(5.2 \pm 0.2) \cdot 10^{-6}$ . For smaller values of  $p$  quantum error correction performs better than using unencoded qubits. For values above this threshold there is no benefit from using quantum error correction at all. Apparently, the benefit becomes larger with increasing algorithm length.

can assure that fault-tolerant quantum error correction will improve the reliability of the quantum computer.

**Correction of decoherence errors** From the results of the  $H^{2k}$ -experiment, we see that the gain is marginal for algorithm lengths of  $k = 1000$  and increases for longer algorithms.<sup>32</sup> This shows that long algorithms will especially profit from quantum error correction, while for rather short algorithms quantum error correction does work, but the gain is rather marginal. Quantum error correction is especially well suited for the protection of quantum memory over large periods of time. The effort of incorporating fault-tolerant

<sup>32</sup>The maximum gain is 2 for a single qubit, since the fidelity of a maximally noisy state is 0.5 (see equation (2.84))



**Figure 2.62** – Gain from using quantum error correction depending on the single qubit error rate  $p$ . Analytical results for various algorithm lengths with the threshold set to  $5 \cdot 10^{-6}$ . The gain is higher for longer algorithms and the position of the maximum shifts to the left with increasing algorithm length.

quantum error correction is only justified if there is a need to protect the quantum bit for a long time period. Otherwise a non-protected qubit will perform only insignificantly worse. We cannot only tell the position of the threshold but also the position of the optimal working point, the point where the gain is maximal. It can be derived from the condition

$$\frac{dg}{dp} \stackrel{!}{=} 0. \quad (2.136)$$

This point, like the threshold, is independent of the system size  $n$ . Of course, it shifts with the length of the algorithm.

Figure 2.62 shows the theoretical prediction of the gain for different algorithm lengths, assuming a determined threshold of  $5 \cdot 10^{-6}$ . While the maximal gain increases with the algorithm length and the position of the maximum shifts towards smaller error rates, all curves intersect in  $(5 \cdot 10^{-6}, 1)$ .

Our simulation code can now be used to make statements about requirements, limits, and thresholds and to determine optimal working points for quantum error correction for arbitrary quantum algorithms concerning decoherence errors. Yet, we will first use our simulation for the analysis of operational imprecisions, which are often neglected in the analytical work.

**Correction of operational errors** We use the same setup as in the determination of the threshold for decoherence errors. Our test algorithm is just a sequence of  $2k$  Hadamard operations. These operations involve unitary over-rotations as described in section 2.2.1. We compare the unencoded case, where we know that the fidelity decays like (equation (2.88))

$$f(k) = \left( \frac{1 + e^{-\frac{9}{2}\sigma^2 k}}{2} \right)^n, \quad (2.137)$$

with  $k$  being the number of  $H^2$ -iterations,  $n$  the number of qubits and  $\sigma$  the standard deviation of the Gaussian distributed over-rotations.

With the assumption that fault-tolerant quantum error correction will also lead to an effective decrease of operational errors according to<sup>33</sup>

$$\sigma_{\text{eff}} = c\sigma^2, \quad (2.138)$$

we can again look at the gain  $g$ , i.e. the ratio of the fidelity of the encoded case to the fidelity of the unencoded case,

$$g(\sigma) = \left( \frac{1 + e^{-\frac{9}{2}(c\sigma^2)^2 k}}{1 + e^{-\frac{9}{2}\sigma^2 k}} \right)^n. \quad (2.139)$$

Here we choose  $k \in \{100; 1000\}$ . Figure 2.63 shows the result of this evaluation. The threshold for operational errors  $\sigma_{\text{thr}} = 1/c$  can be derived from the fit parameter  $c$ . The numerical simulations give the result  $c_{100} = 23.19 \pm 0.1$  and  $c_{1000} = 23.18 \pm 0.09$  for runs with 100 and 1000  $H^2$ -iterations, respectively. For the threshold this means that  $\sigma$  has to be smaller than

$$\sigma_{\text{thr}} = 0.0431 \pm 0.0002 \quad (2.140)$$

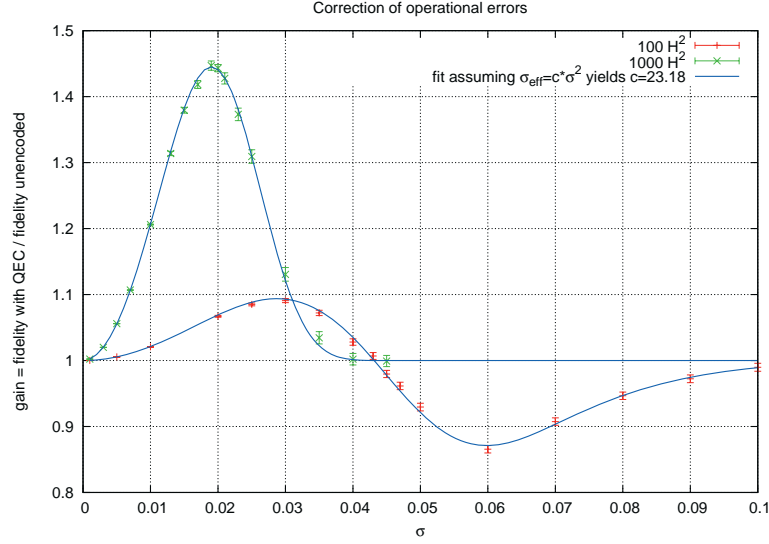
for fault-tolerant quantum error correction to be useful. If  $\sigma$  is larger than  $\sigma_{\text{thr}}$  there is no possibility to get an improvement by using quantum error correction. A Gaussian width of  $\sigma_{\text{thr}} = 0.0431 \pm 0.0002$  corresponds to about  $2.5^\circ$ , which is a rather large value.<sup>34</sup>

Figure 2.63 clearly shows the benefit from quantum error correction for the correction of Gaussian distributed unitary operational imprecisions. For all points above one, running the algorithm with quantum error correction performs better than using unencoded qubits. Even for relatively short algorithm lengths the gain that comes from using quantum error correction is high.<sup>35</sup> Figure 2.63 does not only give us the threshold, but we can also identify

<sup>33</sup>Section B makes clear why this must be the case. Also, assuming a different functional relation does not lead to an adequate fit.

<sup>34</sup>For comparison, ion trap quantum computation has to deal with pulse area errors (and therefore operational over-rotations) of about 0.007(3) [Knill et al., 2008].

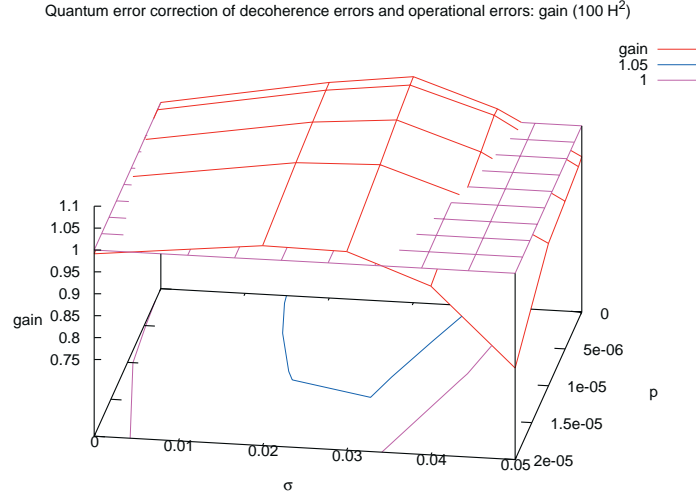
<sup>35</sup>Note, that the maximum gain is 2 in our case, because the completely random state results in a fidelity of 0.5. On the left hand side of figure 2.63 the gain is one, because in the absence of errors, using quantum error correction makes no difference at all, and on the far right it becomes 1 again, when a maximally noisy result (fidelity 0.5) is divided by another maximally noisy result.



**Figure 2.63** – Gain by using quantum error correction to correct for operational imprecisions for a sequence of 100 and a sequence of 1000  $H^2$ -iterations depending on the error parameter  $\sigma$ , i.e. the standard deviation of Gaussian distributed over-rotations. Quantum error correction reduces the error parameter  $\sigma$  to an effective error parameter  $\sigma_{\text{eff}} = c\sigma^2$ . The constant  $c$  is determined from a least squares fit to the data points (using equation (2.139)). It shows that even for short algorithms the benefit from quantum error correction is high. The gain is much higher compared to the gain by protecting against decoherence. A similar threshold can be derived for the correction of operational errors.

the regions, where the benefit from quantum error correction becomes large. The position of the maximal gain can be determined from equation (2.139), once we have determined the threshold  $\sigma_{\text{thr}}$  or the constant  $c$ .

**Correction of decoherence and operational errors** For algorithms of the same length the gain from the correction of operational errors is much higher than the gain from the correction of decoherence errors. That means that quantum error correction is especially well suited for the correction of operational errors. In other words: If operations on qubits have to be done with a previously unattainable precision, it is possible to use logical operations on encoded states to achieve effectively more precise quantum operations than the physical apparatus is able to deliver for a single qubit. Quantum error correction is also well suited for the protection of quantum memory, but the benefit is only large for the protection over long periods of time.



**Figure 2.64** – Assessment of fault-tolerant quantum error correction with Steane’s 7-qubit quantum error correction code for simultaneous decoherence and operational errors. The gain is plotted against the single qubit decoherence probability  $p$  and the operational error parameter  $\sigma$ . Since  $100 H^2$  is a rather short algorithm the gain is not very large. Such a plot would exhibit another structure for other algorithms, where the ratio between idling qubits and qubits which are operated on is different. Nevertheless, this plot shows some interesting characteristics: It tells us something about the “combined” threshold. Even if we are above the single qubit error threshold we can benefit from quantum error correction for a certain  $\sigma$  range (lower edge). But if  $\sigma$  is large (lower right corner), quantum error correction fails, as well as for very small  $\sigma$  values (lower left corner), where it is advisable, not to use quantum error correction. The 1-isoline tell us when there is benefit from using quantum error correction. We have drawn another isoline for  $g = 1.05$  to indicate a good working range for quantum error correction.

With our simulation package we are now able to quantify the effect of fault-tolerant quantum error correction for any parameter range. Thus, we can for example quantify optimal working points in the presence of both decoherence and operational errors. Figure 2.64 shows how Steane’s 7-qubit code performs for an algorithm of short length while suffering from errors of certain magnitudes. Again, the message is that in the presence of operational errors and decoherence errors, one can benefit from using quantum error correction, although the single qubit decoherence error rate might be above threshold. The improvement from the protection against operational errors can outweigh the loss caused by decoherence errors for a certain range of parameters. With numerical simulations we can determine a sort of combined threshold and optimal working ranges.

**Correction of systematic operational errors** So far we have used Gaussian distributed operational imprecisions with zero mean  $\mu$  (equation (2.77)). We can ask the question, what happens if  $\mu \neq 0$ , i.e., an experimental device would systematically over-rotate each operation by a small deviation. For this analysis we set the single qubit decoherence probability  $p$  to zero and concentrate on unitary operational errors only. Again, the algorithm we examine is just a sequence of 200 Hadamard operations. We set up two different scenarios:

1.  $\mu \neq 0, \sigma = 0, p = 0$

Here, there are only systematic errors, while the statistical variance is set to zero. For the unencoded qubit this gives a fully deterministic development of the fidelity (which just oscillates while accumulating an additional rotation angle after each operation). For the quantum error correction case we still have to gather a certain amount of statistics, although  $p$  and  $\sigma$  are set to zero, because the quantum error correction scheme still uses probabilistic measurements.

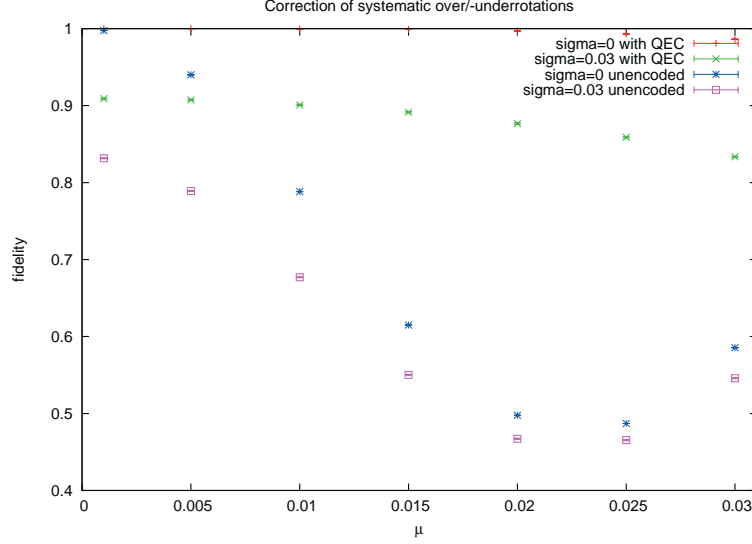
2.  $\mu \neq 0, \sigma = 0.03, p = 0$

This is a realistic case, where the statistical variance is still present, while the system suffers from an additional systematic error. The value for  $\sigma$  has been chosen to lie below the previously determined fault-tolerant threshold at a suitable working point (see figure 2.63).

The results of the numerical simulations are plotted in figure 2.65. Each point is the result of a simulation run of 100  $H^2$ -iterations. The fidelity is plotted against the systematic operational deviation  $\mu$ . For the unencoded cases the fidelity drops rapidly, while for the cases with quantum error correction the fidelity stays high. The increase at values greater than  $\mu = 0.025$  is due to the oscillatory behavior where after 100  $H^2$  operations the over-rotated state rotates closer to the expected value again.

Figure 2.66 gives another view on the same data. Here the gain, i.e. the ratio between the achievable fidelity with and without quantum error correction is plotted against the systematic error  $\mu$ . The plot shows that quantum error correction always gives an improvement when used to correct for systematic errors, independent of any threshold. In a real quantum computation device systematic errors are probably more easily identified and compensated for than going to quantum error correction schemes. Nevertheless, if some kind of quantum computing architecture exhibits systematic over-rotational errors, quantum error correction can overcome this obstacle. Systematic over-rotational errors are not an invincible problem for quantum error correction, as quantum error correction deals with this sort of problem quite well.

**Quantum teleportation with fault-tolerant quantum error correction** An evaluation of the performance of quantum error correction for a realistic scenario, namely quantum teleportation [Bennett et al., 1993], has been done using the JUMPIQCS simulation package. We analyze the quantum teleportation algorithm that is shown in figure 2.67. The main resource used in quantum teleportation is the entangled Bell pair. As such, our analysis is



**Figure 2.65** – Use of quantum error correction to correct for systematic over-rotations. Fidelities with and without quantum error correction for the parameters  $\mu \neq 0$ ,  $\sigma = 0$ ,  $p = 0$  and  $\mu \neq 0$ ,  $\sigma = 0.03$ ,  $p = 0$ . The use of quantum error correction always gives better results than dealing with uncoded qubits.

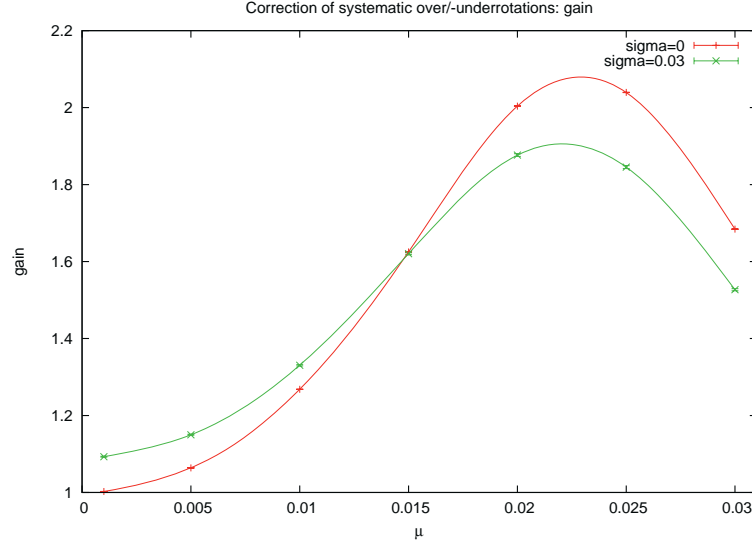
an evaluation of the preservation of quantum entanglement between two encoded logical qubits. Quantum teleportation uses only gates of the Clifford group, so that we can use fully fault-tolerant encoded quantum gate operations.

We initialize the original qubit in the equal superposition state  $2^{-1/2}(|0\rangle + |1\rangle)$  and compare this to the resulting teleported state. A first analysis of the quantum teleportation algorithm shows that the algorithm itself is so short, that quantum error correction does not give any significant benefit at all. Theoretical descriptions of quantum teleportation usually do not consider any errors. What is usually neglected is that the time during which the Bell state qubits are being spatially separated, as well as the time to send the classical information, can become significantly large. We name these times travel-time and message-time.

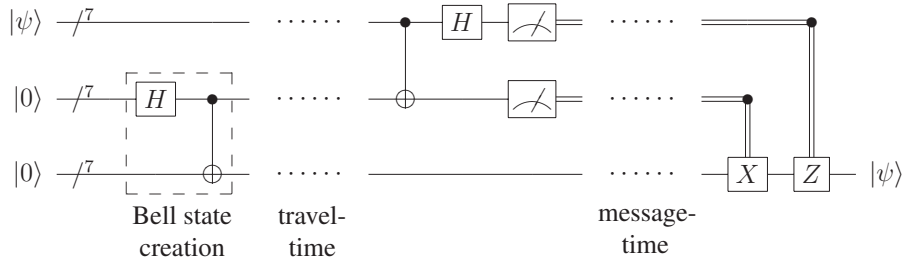
In a real world experiment the qubits, the original one as well as the qubits of the Bell state, will suffer mainly from decoherence errors. Since no operations have to be done during the travel-time and the message-time (except those of the quantum error correction steps), this is mainly a problem of stabilizing quantum memory. The effect of protection against operational errors is negligible.

It takes a certain time to move the target qubit from the initial to the final position. Going to quantum error correction, there are two different scenarios conceivable. The travel-time is divided into discrete timesteps.

### 2.3. SIMULATION OF QUANTUM ERROR CORRECTION



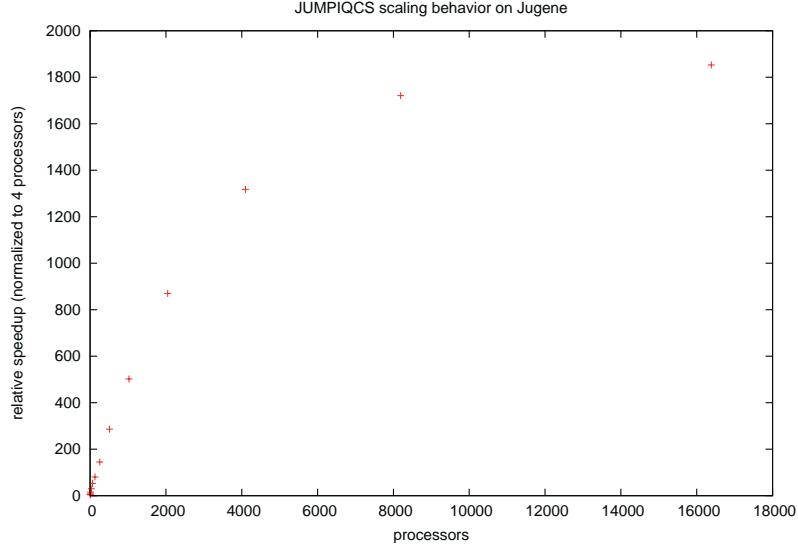
**Figure 2.66** – Gain by using quantum error correction to correct for systematic over-rotations. For better visualization cubic splines have been drawn through the data points. Quantum error correction always gives a benefit when used to correct for systematic errors.



**Figure 2.67** – Quantum teleportation circuit: After the creation of the Bell state, both qubits of the Bell pair are spatially separated and used as a resource to transmit quantum information over a classical channel. The separation of the qubits takes some time (travel-time) as well as the transmission of the classical information about the measurement results (message-time). During these times the (logical) qubits should be protected by quantum error correction.

1. In the first scenario the correction measures are taken in between each timestep. Although each correction step takes many timesteps, the qubit does not move closer to the final destination during correction. This can be thought of as flying qubits which are relayed from station to station, where at each station a quantum error correction step can be performed.





**Figure 2.68** – Relative speedup for the JUMPIQCS code using quantum error correction for a system size of 26 qubits. The processors have been accessed in Virtual Node mode on the JUGENE system, i.e., each processor is assigned to one MPI task.

2. The second case describes a qubit, where the whole apparatus is moving, so that the correction steps can be done simultaneously while the qubit moves closer to the destination. Here quantum error correction will yield better results.

We did a comparison using the first scenario for the parameter set  $p = 10^{-6}$ ,  $\sigma = 0$ , travel-time = 10000 and message-time = 1000, each for the unencoded case and for the protected case using Steane’s 7-qubit code. The non-encoded case gives a mean fidelity of  $0.9796 \pm 0.0005$  using  $m = 100000$  statistical iterations. For the quantum error correction case that uses 3 logical qubits and that has to work on a state vector of a 26 qubit system, we were not able to gather such high statistics. Nevertheless, on the JUGENE system using 2048 processors and a runtime of about 300 hours we could gather  $m = 100$  statistical iterations for this large system and very long algorithm length (considering the travel-time of 10000 timesteps and a correction step of about 1500 timesteps each time). The result is that all potential errors during these  $m = 100$  iterations could be perfectly corrected, so that the resulting fidelity stays constantly at one. However, this statement has to be regarded cautiously. Apparently, higher statistics may be gathered, once further computational resources are available, and so we focus on scalability and efficiency of the simulation. After applying basic compiler optimizations along with platform specific adjustments, a performance analysis reveals an almost linear relative speedup up to 4096 processors before the curve begins to flatten (see figure 2.68). More simulation runs with various sets of parameters

will be performed in the future.

In summary, we can say that quantum error correction can improve the teleportation process by protecting an entangled encoded Bell pair. However, for relatively short travel- and message-times the fidelity of unprotected qubits stays close to one, so that quantum error correction is simply unnecessary. For longer travel- and message-times quantum error correction becomes more and more important for the protection of idling memory qubits as expected. This aspect is often neglected in theoretical analyses of teleportation and must be examined further.

**Grover’s search algorithm using fault-tolerant quantum error correction** The JUMPIQCS package has also been used to analyze the behavior of Grover’s search algorithm (see section 2.1.2.2) when including quantum error correction. The details can be found in [Peschina, 2008]. Several approaches have been studied:

1. The first approach is motivated by the fact, that the ancilla qubit plays a major role in Grover’s algorithm (see sections 2.1.2.2 and 2.2.4). Therefore, only this ancilla qubit is encoded and protected by Steane’s 7-qubit code, while the other qubits remain unencoded. With this approach, we can handle a much larger search space, since it requires only  $\log N + 7 + 5$  qubits to deal with a search space of  $N$  elements;  $\log N$  qubits are needed for the storage of the search indices, 7 qubits for the storage of the encoded ancilla qubit and 5 additional ones for the syndrome extraction of the quantum error correction scheme. On the JUGENE system this limits the search space to  $N = 2^{28}$  elements.
2. The second approach uses a full encoding of all qubits of the system, i.e., for the same search space of size  $N$  we need  $7 \log N + 7 + 5$  qubits. With the current status of the JUGENE system<sup>36</sup> the memory limit would be reached at  $N = 2^4$ .

Unfortunately, it turns out that “it is not possible to correct the whole algorithm by only stabilizing the ancilla qubit” [Peschina, 2008]. This result refutes our initial assumption that it might be enough to protect the ancilla qubit to gain a benefit from quantum error correction. Therefore, only a full encoding of all involved qubits can lead to an improvement.

Concerning the analysis of the fully encoded case, where all qubits are encoded using Steane’s 7-qubit code, one has to specify the implementation of the oracle (figure 2.30). Since a fully fault-tolerant  $\pi/8$ -gate has not yet been implemented into JUMPIQCS, that would be required to decompose the  $C^n$ NOT operation into elementary single- and two-qubit gates, we have decided to take another approach. This is important for the conclusions drawn from this analysis, so we emphasize this issue here. The procedure is to do an ideal decoding step previous to the  $C^n$ NOT operation followed by an ideal encoding step. Ideal means that the decoding and encoding is done completely error free. Nevertheless this

<sup>36</sup>JUGENE is currently being upgraded. The 16 rack system with 65536 processors and 32 TB of memory will be upgraded to 72 racks and a total memory size of 144 TB.

method involves a sort of passive error correction. Although no correction step is done, a projection of spurious amplitudes back into the code space happens with each decoding/encoding step. This might have a non-negligible effect on the outcome. The effect of this passive stabilization has been estimated by comparing this to the case with active stabilization steps, i.e. syndrome extraction and correction. The differences detected are marginal, so we assume that a great part of the improvement can be related to the passive stabilization. Simulation runs for a fully encoded system have been done for a search space of  $N = 4$ , which equals to a total system size of 26 qubits.

The conclusion drawn from this analysis is that fault-tolerant quantum error correction is able to improve the performance of Grover's search algorithm. But this is not possible by stabilizing the ancilla qubit only. A full encoding of all qubits is required, thus reducing the practical problem sizes enormously. The benefit from using quantum error correction applied to Grover's search algorithm is marginal for the cases that have been examined. This is not surprising, because a Grover search of  $N = 4$  elements involves just one Grover iteration to find the solution (see equation (2.43)). Thus, the algorithm length is minimal. As we stated previously, quantum error correction gives more benefit when going to longer algorithms. For larger search spaces more Grover iterations have to be done and so the algorithm length increases, but for now, we are limited by the available memory resources, so that we cannot study large search spaces with a full encoding of all qubits that are involved. For details of this analysis refer to [Peschina, 2008].

### 2.3.3.3 Conclusion

For the cases that we have studied, the simulation results tell us how good the performance of a useful quantum computer has to be. Even with the limited resources of the quantum computation devices of the near future it might be helpful to use quantum error correction both for the storage of quantum information as well as for doing computations.

We have determined a threshold for fault-tolerant quantum computation which is  $p_{\text{thr}} = (5.2 \pm 0.2) \cdot 10^{-6}$ . Of course, an error rate of  $5 \cdot 10^{-6}$  is beyond the scope of today's quantum computation devices and for the near future this is surely an ambitious goal to endeavor, but there is no physical limitation that prohibits to achieve this threshold.<sup>37</sup> State-of-the-art ion trap quantum computation can currently perform initialization, readout and single qubit operations with error rates of the order  $10^{-3}$  and two qubit operations with error rates of  $10^{-2}$  to  $10^{-1}$  [Häffner et al., 2008].

If the single qubit error probabilities are above the threshold for fault-tolerant quantum error correction, there would be no sense in doing quantum error correction at all, because all attempts will necessarily introduce more errors than quantum error correction can correct.

---

<sup>37</sup>For comparison, in classical computation the error rates today are considerably (orders of magnitudes) lower than  $10^{-6}$ .

Yet, this statement can be relaxed considering the ability of quantum error correction to protect against operational imprecisions by doing operations on encoded qubits.

Assuming that one can reach the threshold, fault-tolerant quantum error correction is especially well suited for the protection of quantum bits over long periods of time, e.g. in quantum memory. For the protection over a short timeframe ( $\lesssim 1000$  gate operations) we cannot recommend using quantum error correction without reservation, because the benefit does not justify the cost, although using fault-tolerant quantum error correction does not degrade the fidelity of the protected qubit. Nevertheless, the improvement by using quantum error correction is only marginal.

The simulation results suggest that quantum error correction is especially well suited for the correction of operational imprecisions. There is also a fault-tolerant threshold for this kind of errors, that we can quantify as  $\sigma_{\text{thr}} = 0.043$ . With this we can eliminate doubts about the performance of quantum error correction under unitary over-rotational errors. In disagreement with [Hill and Viamontes, 2008], who made the statement that quantum error correction was not able to correct unitary over-rotations, and in consequence a realistic quantum computer would never be able to run Shor’s algorithm, we could show that quantum error correction is indeed well suited for the correction of operational unitary over-rotation errors.<sup>38</sup>

Another result of our simulations is that quantum error correction can also be used to protect against systematic over-rotational errors. This is a problem that appears in many setups, for example it is prominent in NMR quantum computation devices.<sup>39</sup> At least in our examples studied, we show that quantum error correction can be used to correct systematic over-rotations independent of any threshold.

Although we have only analyzed a limited set of algorithms using JUMPIQCS, we have developed a simulation package that can, in principle, be used to make statements about requirements of future quantum computer architectures, to calculate limits and thresholds and that can be applied to arbitrary quantum algorithms to determine optimal working points for quantum error correction.

Remember that all statements apply to the general assumptions of our error model. For a specific device in the lab one may tailor quantum error correction schemes to protect against specific errors, so that this might allow a benefit from quantum error correction even at higher error rates. Also going to higher levels of concatenation as described in section 2.3.3 can also improve the performance significantly, provided that the necessary amount of qubits will become available.

---

<sup>38</sup>They do not seem to have used a fully fault-tolerant approach, which probably explains their contradicting result.

<sup>39</sup>Yet, NMR quantum computer devices will probably not scale up to large enough system sizes [Hughes, 2004] to incorporate quantum error correction. Here other techniques to correct for systematic errors are used, e.g. composite pulses.

Note, that there are also other paradigms of quantum computation beyond the quantum gate model: Adiabatic quantum computation [Farhi et al., 2000] for example seems to incorporate some intrinsic stability against errors. Another theoretical proposal is topological quantum computation that uses anyons for quantum computation [Kitaev, 2003]. But up to now this is purely theoretical and has not yet been realized in the lab. The most common quantum computational model today still relies on the quantum gate model. For this model and the analyzed cases, we have shown under which conditions quantum error correction can give a benefit.

## Chapter 3

# Dynamic Simulations of Ion Trap Quantum Computers

...we never experiment with  
just one electron or atom or  
small molecule.

*(Erwin Schrödinger, 1952)*

### 3.1 Theory of Ion Trap Quantum Computation

At this time the pursuit for the best quantum computing technology is not yet decided. Although various technologies are competing and each has its own advantages and drawbacks [Hughes, 2004], one of the most advanced candidates for realizing a scalable quantum computer is ion trap quantum computation [Monroe et al., 1995; Wineland et al., 1998; Leibfried et al., 2003a]. Certainly, it fulfills the criteria required for the realization of a quantum computation device [DiVincenzo, 2000]:

- It has well defined representations of qubits which are scalable.
- Fiducial initial states can be prepared.
- The decoherence time is long compared to the gate operation time.
- A universal set of quantum gates can be realized.
- A qubit specific measurement of the output is possible.

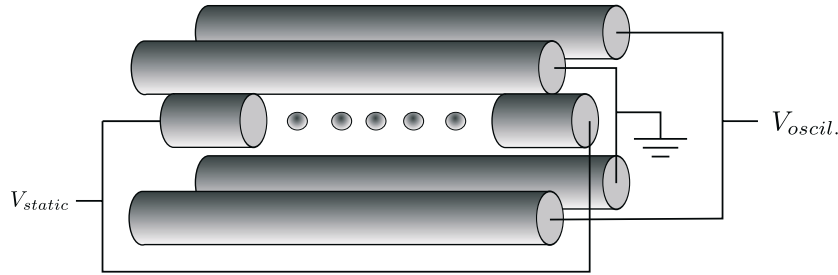
For comparison, table 3.1 gives an estimate about the decoherence times of various technologies (taken from [Nielsen and Chuang, 2000]).<sup>1</sup>

---

<sup>1</sup>The decoherence time indicates the time period until a quantum state has completely decayed into a classical state. It should not be confused with the threshold determined in section 2.3.3.2, which indicates how many operations can be done until a single qubit error happens.

System	$\tau_Q$ [s]	$\tau_{op}$ [s]	$n_{op}$
Nuclear spin	$10^{-2} - 10^8$	$10^{-3} - 10^{-6}$	$10^5 - 10^{14}$
Electron spin	$10^{-3}$	$10^{-7}$	$10^4$
Ion trap ( $\text{In}^+$ )	$10^{-1}$	$10^{-14}$	$10^{13}$
Electron - Au	$10^{-8}$	$10^{-14}$	$10^6$
Electron - GaAs	$10^{-10}$	$10^{-13}$	$10^3$
Quantum Dot	$10^{-6}$	$10^{-9}$	$10^3$
Optical cavity	$10^{-5}$	$10^{-14}$	$10^9$
Microwave cavity	$10^0$	$10^{-4}$	$10^4$

**Table 3.1** – Rough estimates for decoherence times  $\tau_Q$ , operation times  $\tau_{op}$  for a quantum operation, and the maximum number of operations  $n_{op} = \tau_Q/\tau_{op}$ . As ion trap quantum computation exhibits good decoherence times, it is a good candidate for handling quantum information.



**Figure 3.1** – Sketch of linear Paul trap. Ions are confined to the trap axis by an oscillatory potential. The end caps provide static confinement along trap axis.

We will give a brief overview of how qubits are realized, how initialization, control and readout on ion trap quantum computers work and how the mentioned DiVincenzo criteria are met. For comprehensive reviews refer to [Wineland et al., 1998; Leibfried et al., 2003a; Häffner et al., 2008].

**Physical apparatus** An ion trap quantum computer uses a linear Paul trap [Paul, 1990] to confine charged particles in three dimensions. Since a confinement with static electric fields in three dimensions is not possible according to Earnshaw’s theorem [Earnshaw, 1842], the particles are trapped by oscillating radiofrequency fields. Usually four cylindrical rod electrodes are used (figure 3.1) where diagonally opposite rods are driven by an oscillating voltage. This gives rise to a ponderomotive potential that confines the ions in radial direction. Additionally, DC electrodes at both ends of the trap confine the motion of the ions in axial direction. For the equations of motion refer to [Wineland et al., 1998]. For our studies it is sufficient to realize that the ions are confined by a harmonic pseudopotential in the radial direction and a static harmonic potential in the axial direction. The trap

frequency in radial direction is chosen to be much larger than in axial direction forcing a collection of cooled ions to line up along the axis of the trap. The non-equidistant spacings of the equilibrium positions are defined by a balance between mutual Coulomb repulsion and external force (see [Sasura and Buzek, 2002] for details). Nevertheless, to a good approximation, the result is a linear string of ions in a harmonic potential, where the ions can be addressed individually by well tuned and directed laser beams. In our simulations we will concentrate on the axial center of mass motion of the linear ion string for which the harmonic approximation is valid.

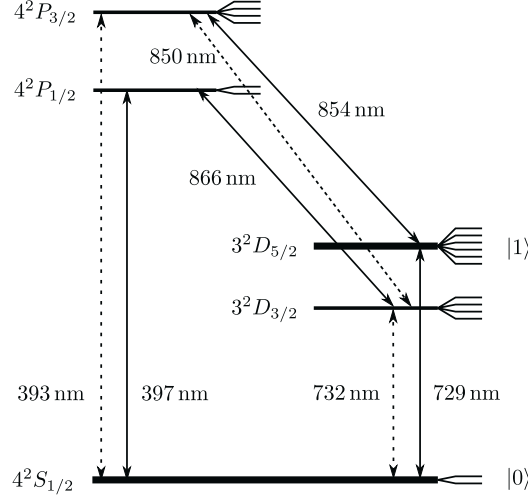
**Atomic structure of qubit ions** Quantum computation with trapped ions has been realized with various choices of ions, e.g. beryllium [Monroe et al., 1995; Wineland et al., 1998] or calcium [Nägerl et al., 1998; Häffner et al., 2008]. The main criterion for a good candidate is a long decoherence time. This can be achieved by different techniques. While the NIST group in Boulder [Monroe et al., 1995; Wineland et al., 1998] uses  $\text{Be}^+$ -ions, where the qubits are encoded via hyperfine transitions and qubits are manipulated via Raman transitions, the ion of choice that the Innsbruck group uses is  $^{40}\text{Ca}^+$  with metastable states and optical transitions [Nägerl et al., 1998; Häffner et al., 2008]. More recently, experiments with  $^{43}\text{Ca}^+$  are done [Benhelm, 2008], which use the ground state hyperfine structure. This has the advantage of about a thousand times longer decoherence times, and it is less susceptible to environmental influences. As an example, figure 3.2 shows the level scheme of the levels populated during an experiment with  $^{40}\text{Ca}^+$  [Gulde, 2003]. The levels  $S_{1/2}$  and  $D_{5/2}$  are associated with the logical states  $|0\rangle$  and  $|1\rangle$ , respectively. The metastable state  $D_{5/2}$  has a long lifetime of about 1 s [Gulde, 2003]. Optical pumping on the  $S_{1/2} \leftrightarrow P_{1/2}$  transition is used for Doppler cooling of the ion and for the initialization in the  $S_{1/2}$  ground state. The existence of another metastable state,  $D_{3/2}$ , requires another pumping laser at 866 nm. A laser at 854 nm can be used to repump any  $D_{5/2}$  population to the  $S_{1/2}$  state.

The measurement of the qubits at the end of a quantum algorithm can be performed with a fidelity of 99.9% [Gulde, 2003] by measuring the fluorescence while driving the  $S_{1/2} \leftrightarrow P_{1/2}$  and  $D_{3/2} \leftrightarrow P_{1/2}$  transition at 397 nm and 866 nm. If the ion is in the  $P_{1/2}$  state, fluorescence will be detected, whereas the  $D_{5/2}$  state will remain dark.

So far, the DiVincenzo criteria for well defined qubits, initialization, readout and coherence are fulfilled. Next, we will explain how explicit control over the system for doing quantum operations can be exercised. In the following we will treat the ion trap system as harmonically trapped two-level atoms. We first describe the situation for a single ion and later on extend the model to the description of multiple ions, while limiting the model to the collective center-of-mass motion only.

**Hamiltonian of the ion trap system with laser-ion interaction** The simplified model of a single two level system in a harmonic potential addressed by a laser field is described by





**Figure 3.2** – Atomic energy levels of  $^{40}\text{Ca}^+$  with Zeeman substructure. The transitions indicated by solid arrows are driven by laser radiation during the experiment. The levels  $S_{1/2}$  and  $D_{5/2}$  are associated with the logical states  $|0\rangle$  and  $|1\rangle$ . We use the standard atomic level notation  $n^{2S+1}L_J$ , where  $n$  is the principal quantum number,  $S$  is the spin angular momentum,  $L$  is the orbital angular momentum and  $J$  is the total angular momentum.

the Hamiltonian [Leibfried et al., 2003b]

$$H(t) = \underbrace{\frac{p^2}{2m} + \frac{m}{2}\omega_t^2 x^2}_{\text{harmonic oscillator}} + \underbrace{\frac{1}{2}\hbar\omega_a\sigma_z}_{\text{spin part}} + \underbrace{\frac{1}{2}\hbar\Omega(\sigma^+ + \sigma^-)(e^{i(kx - \omega_t t + \phi)} + e^{-i(kx - \omega_t t + \phi)})}_{\text{laser-ion interaction}}, \quad (3.1)$$

with  $p$  and  $x$  being the momentum and position operator and  $m$  being the mass of the ion. The trap frequency in axial direction is denoted by  $\omega_t$ , and  $\omega_a$  is the atomic transition frequency. The symbol  $\sigma_z$  denotes the Pauli- $Z$  matrix and  $\sigma_{\pm}$  are the atomic raising and lowering operators, the linear combination of the Pauli matrices  $\sigma_{\pm} = \frac{1}{2}(\sigma_x \mp i\sigma_y)$ .<sup>2</sup> The strength of the laser-ion interaction is described by the coupling constant  $\Omega$ , which is called the Rabi frequency. The parameters of the laser radiation are the wave number  $k$ , the laser frequency  $\omega_t$ , and the phase of the laser  $\phi$ .

While this Hamiltonian can be numerically solved, as we will explain in section 3.2, it is beneficial for the understanding of the system to have a look at an approximative analytical description. We follow the work presented in [Leibfried et al., 2003b]. The ion's external

<sup>2</sup>Note, that  $\sigma_+$  is defined with a minus and vice versa. This is due to our previous definition (equation (2.1)) of  $|0\rangle \doteq \begin{pmatrix} 1 \\ 0 \end{pmatrix}$  and  $|1\rangle \doteq \begin{pmatrix} 0 \\ 1 \end{pmatrix}$ , so that  $\sigma_+ |0\rangle = |1\rangle$  and  $\sigma_- |1\rangle = |0\rangle$ .

degrees of freedom, i.e. harmonic oscillations, can be described quantum mechanically by introducing the raising and lowering operators  $a^\dagger$  and  $a$  with

$$x = \sqrt{\frac{\hbar}{2m\omega_t}} (a^\dagger + a), \quad (3.2)$$

$$p = i\sqrt{\frac{\hbar m\omega_t}{2}} (a^\dagger - a), \quad (3.3)$$

and  $[a, a^\dagger] = 1$ , so that

$$H_0 = \hbar\omega_t(a^\dagger a + \frac{1}{2}) + \frac{1}{2}\hbar\omega_a\sigma_z. \quad (3.4)$$

The expression  $\sqrt{\hbar/(2m\omega_t)}$  describes the size of the ground state wave function and we introduce the Lamb-Dicke parameter

$$\eta = k\sqrt{\frac{\hbar}{2m\omega_t}}, \quad (3.5)$$

which describes the relation between the laser wavelength and the size of the ground state wave function.<sup>3</sup> With this notation the laser-ion interaction term of the Hamiltonian can be written as

$$H_1 = \frac{1}{2}\hbar\Omega(\sigma_+ + \sigma_-) \left( e^{i(\eta(a^\dagger + a) - \omega_t t + \phi)} + e^{-i(\eta(a^\dagger + a) - \omega_t t + \phi)} \right). \quad (3.6)$$

Several approximations are applied<sup>4</sup> to simplify this expression:

1. Assuming we are in the Lamb-Dicke regime  $\eta \ll 1$ , i.e., that the extension of the atomic wave packet is much smaller than the wavelength of the transition, we can make a first order expansion in  $\eta$  and get

$$H_1 = \hbar\Omega(\sigma_+ + \sigma_-) \left[ (1 + i\eta(a^\dagger + a))e^{i(-\omega_t t + \phi)} + (1 - i\eta(a^\dagger + a))e^{-i(-\omega_t t + \phi)} \right] + O(\eta^2). \quad (3.7)$$

2. Going to the interaction picture (or Dirac picture) with  $U_0 = e^{-\frac{i}{\hbar}H_0 t}$  and  $H_I = U_0^\dagger H_1 U_0$ , we get the time-dependent operators

$$\sigma_+(t) = \sigma_+ e^{i\omega_a t}, \quad (3.8)$$

$$\sigma_-(t) = \sigma_- e^{-i\omega_a t}, \quad (3.9)$$

$$a^\dagger(t) = a^\dagger e^{i\omega_t t}, \quad (3.10)$$

$$a(t) = a e^{-i\omega_t t}. \quad (3.11)$$

Doing the rotating wave approximation, i.e. neglecting the rapidly oscillating terms [Leibfried et al., 2003b], the interaction Hamiltonian  $H_I$  can be further simplified.

<sup>3</sup>If the direction of the laser beam is at an angle  $\beta$  to the oscillation axis, the Lamb-Dicke parameter is defined as  $\eta = k \cos \beta \sqrt{\hbar/(2m\omega_t)}$ .

<sup>4</sup>These approximations can be applied in any particular order.

3. Assuming that the laser is tuned to resonant transitions, i.e.

$$\omega_l - \omega_a = n\omega_t, \quad (3.12)$$

with  $n \in \{-1, 0, 1\}$  only (neglecting higher order transitions), the interaction Hamiltonian  $H_I$  can be simplified to three simple expressions.

The final interaction Hamiltonian is given by the following equations:

1. For  $\omega_l = \omega_a$ , i.e., when the laser is tuned to the atomic transition frequency:

$$H_I^{\text{car}} = \frac{1}{2}\hbar\Omega(\sigma_+e^{i\phi} + \sigma_-e^{-i\phi}). \quad (3.13)$$

This is called a carrier transition.

2. For  $\omega_l = \omega_a - \omega$ :

$$H_I^{\text{rsb}} = \frac{1}{2}\hbar\eta\Omega(a\sigma_+e^{i\hat{\phi}} + a^\dagger\sigma_-e^{-i\hat{\phi}}), \quad (3.14)$$

with  $\hat{\phi} = \phi + \frac{\pi}{2}$ . This is called the first red sideband transition. The atom absorbs the laser radiation, which is tuned to the atomic transition frequency reduced by the energy of a vibrational quantum. The atom interacts with the phonon mode and changes the vibrational quantum number. This Hamiltonian is also well known from cavity quantum electrodynamics as the Jaynes-Cummings Hamiltonian [Jaynes and Cummings, 1963].

3. For  $\omega_l = \omega_a + \omega$ :

$$H_I^{\text{bsb}} = \frac{1}{2}\hbar\eta\Omega(a^\dagger\sigma_+e^{i\hat{\phi}} + a\sigma_-e^{-i\hat{\phi}}), \quad (3.15)$$

with  $\hat{\phi} = \phi + \frac{\pi}{2}$ . Here the excitation process is accompanied by an increase of the vibrational quantum number and it is termed the first blue sideband excitation. This Hamiltonian is also named anti-Jaynes-Cummings Hamiltonian.

Figure 3.3 visualizes these three cases.

**Single qubit gates** If the laser is driven with the resonance frequency of the atomic transition,  $\omega_l = \omega_a$ , the qubit state of the illuminated ion is rotated according to

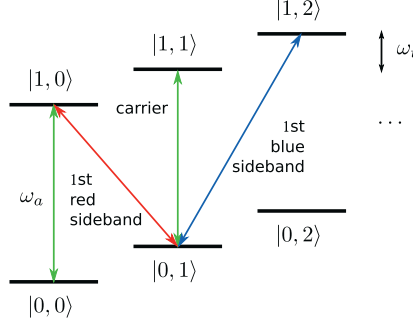
$$R^{\text{car}}(\theta, \phi) = e^{-\frac{i}{\hbar}H_I^{\text{car}}t} = e^{-i\frac{\theta}{2}(\sigma_+e^{i\phi} + \sigma_-e^{-i\phi})}, \quad (3.16)$$

where the rotation angle  $\theta$  is given by  $\Omega t$  and the axis of rotation is defined by  $\phi$ . If we choose  $\phi = 0$ , this results in

$$R(\theta, 0) = e^{-i\frac{\theta}{2}\sigma_x} = R_x(\theta), \quad (3.17)$$

which is a rotation around the  $x$ -axis. Choosing  $\phi = \pi/2$  gives

$$R(\theta, \frac{\pi}{2}) = e^{-i\frac{\theta}{2}\sigma_y} = R_y(\theta), \quad (3.18)$$



**Figure 3.3** – Energy levels for a two-level ion in a harmonic potential. The notation  $|q, n\rangle$  is used for a qubit with the internal state  $|q\rangle$  and a vibrational quantum number of  $n$ . A carrier transition changes just the occupation of the internal atomic levels. A blue sideband excitation simultaneously increases the vibrational phonon number, whereas a red sideband excitation decreases the phonon number.

a rotation around the  $y$ -axis. Rotations around the  $z$ -axis can be decomposed into rotations around the other two axes.<sup>5</sup> Thus, any arbitrary single qubit operation can be done with ion trap quantum computation.

**Rabi frequencies** The coupling strengths will be varying depending on the detuning  $\omega_l - \omega_a$ , and therefore on the coupling of certain internal and motional states according to the interaction Hamiltonian. We use the notation  $|q, n\rangle$  for a qubit with the internal state  $|q\rangle$  and vibrational quantum number  $n$ . The transitions from  $|0, n\rangle$  to  $|1, n+s\rangle$  take place with the Rabi frequencies [Leibfried et al., 2003b]

$$\begin{aligned} \Omega_{n,n+s} &= \Omega_{n+s,n} = \Omega | \langle n+s | e^{i\eta(a^\dagger+a)} | n \rangle | \\ &= \Omega e^{-\frac{\eta^2}{2}} \eta^{|s|} \sqrt{\frac{n_{<}!}{n_{>}!}} L_{n_{<}}^{|s|}(\eta^2), \end{aligned} \quad (3.19)$$

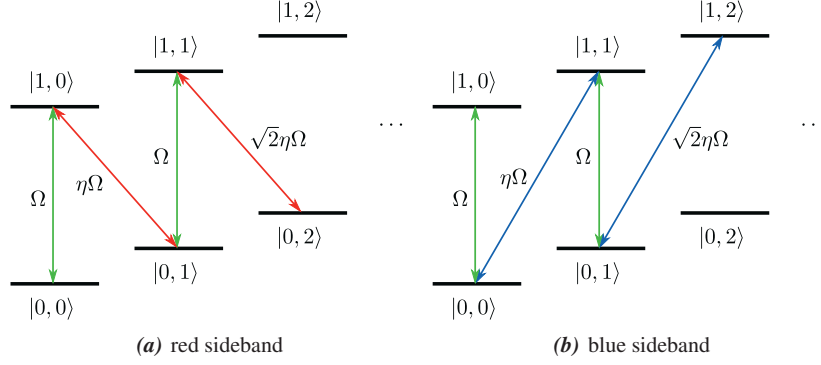
where  $n_{<}$  ( $n_{>}$ ) is the lesser (greater) of  $n+s$  and  $n$ , and  $L_n^\alpha(X)$  is the generalized Laguerre polynomial

$$L_n^\alpha(X) = \sum_{m=0}^n (-1)^m \binom{n+\alpha}{n-m} \frac{X^m}{m!}. \quad (3.20)$$

The interesting cases for ion trap quantum computation are:

$$\Omega_{n,n} = \Omega e^{-\frac{\eta^2}{2}} \approx \Omega, \quad (3.21)$$

<sup>5</sup>The error model introduced in section 2.2.1 uses a decomposition of general rotations into rotations around a different pair of axes, namely the  $y$ -axis and the  $z$ -axis. Therefore, the error model does not directly resemble errors occurring in ion trap quantum computation, where each rotation is decomposed into rotations around the  $x$ -axis and the  $y$ -axis.



**Figure 3.4** – Coupling strengths of carrier and sideband transitions. Sideband transitions are a factor of  $\eta$  (equation (3.5)) weaker than carrier transitions. Furthermore, the coupling of the sideband transitions are dependent on the phonon number.

and

$$\Omega_{n,n+1} = \sqrt{n+1}\eta\Omega e^{-\frac{\eta^2}{2}} \approx \sqrt{n+1}\eta\Omega. \quad (3.22)$$

This means that the sideband transitions are a factor of  $\eta$  weaker and therefore slower than the carrier transition. Furthermore, the carrier transition coupling is independent of the number of phonons, whereas the strength of the sideband transitions are sensitive to the phonon number (see figure 3.4). This is important for the realization of the composite pulse CNOT gate (see below).

We have assumed that the laser will be exactly on resonance and that Rabi oscillations take place that change the population of the two resonant levels periodically. However, if there is a detuning  $\delta$  from any resonant transition, this will result in incomplete population transfers with lower amplitudes

$$A_\delta = \frac{\Omega_{n,n+s}^2}{\delta^2 + \Omega_{n,n+s}^2} \quad (3.23)$$

and higher frequencies

$$f_\delta = \sqrt{\delta^2 + \Omega_{n,n+s}^2} \quad (3.24)$$

[Gulde, 2003].

**Sideband rotations** Rotations can also be done on the sideband, i.e., driving the laser with the atomic frequency plus or minus the energy of the vibrational mode. A rotation on the red sideband is described by

$$R^{\text{rsb}}(\theta, \phi) = e^{-\frac{i}{\hbar} H_I^{\text{rsb}} t} = e^{-i\frac{\theta}{2}(\sigma_+ a e^{i\phi} + \sigma_- a^\dagger e^{-i\phi})}, \quad (3.25)$$

where the rotation angle  $\theta$  is now defined by  $\theta = \sqrt{n+1}\eta\Omega t$ , and a blue sideband rotation can be written accordingly as

$$R^{\text{bsb}}(\theta, \phi) = e^{-\frac{i}{\hbar} H_I^{\text{bsb}} t} = e^{-i\frac{\theta}{2}(\sigma_+ a^\dagger e^{i\phi} + \sigma_- a e^{-i\phi})}. \quad (3.26)$$

**Two-qubit operations (CNOT gate)** The remaining DiVincenzo criterion that must be met is the possibility to do any unitary quantum operation by having a universal set of gates (see section 2.1.1). It has already been shown how single qubit rotations can be realized. For universal quantum computation it is sufficient to show that an ion trap quantum computer can perform a CNOT operation as well. Ion trap quantum computation became popular, when [Cirac and Zoller, 1995] proposed how to perform a CNOT gate with ions in a Paul trap.

First, we will briefly outline the original proposal by [Cirac and Zoller, 1995] and afterwards we will show how to realize a CNOT gate with “composite pulses”.

In the Cirac-Zoller scheme the ions are coupled through a common vibrational phonon mode. Interactions between any two of the ions within the string are possible by transferring the quantum state information of any qubit to the vibrational mode of the so called phonon bus. As this vibration is common to the whole string of ions, two (distant) qubits in the string can interact via this phonon bus. The coupling between electronic and motional degrees of freedom can be accomplished by driving sideband transitions as described.

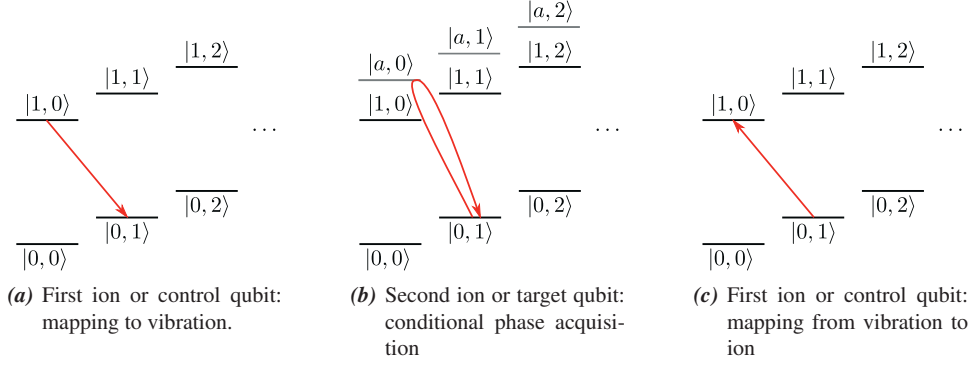
The precondition for the Cirac-Zoller CNOT gate is a well defined vibrational state. Usually the ions are cooled to the quantum motional ground state, i.e., the vibrational quantum number is 0. This scheme also requires an auxiliary third level that can be addressed individually (see figure 3.5). The Cirac-Zoller scheme suggests the following steps:

1. A red sideband  $\pi$ -pulse on the first ion will move any population from the  $|1, 0\rangle$  state to the  $|0, 1\rangle$  state.<sup>6</sup> If the ion was in the  $|0, 0\rangle$  state the laser pulse would not change its state. The effect is a mapping of the internal state of the ion to the external motional degree of freedom.
2. The motional state is coupled to the target ion by another red sideband pulse on the second ion, but this time a  $2\pi$ -pulse is done between the  $|0, 1\rangle$  state and an auxiliary level  $|a, 0\rangle$ . While the state  $|0, 1\rangle$  acquires a phase<sup>7</sup> of  $-1$ , the states  $|0, 0\rangle$ ,  $|1, 0\rangle$  and  $|1, 1\rangle$  are not affected, because there are no levels to which they can couple with this transition energy.
3. Another red sideband  $\pi$ -pulse on the first ion, maps the vibrational state back to the internal state of the first ion.

---

<sup>6</sup>Depending on the phase  $\phi$ , i.e. on the rotation axis, the state may acquire a global phase, e.g.  $-i$  for a rotation around the  $x$ -axis ( $\phi = 0$ )

<sup>7</sup>This creates a significant local phase difference.



**Figure 3.5** – Cirac-Zoller scheme for a controlled phase gate. The internal state of the control ion is mapped to the vibrational quantum state (a), the target qubit acquires a phase flip (b) conditioned on the vibrational mode and the motion is mapped back to the control ion (c). Enclosed between two  $\pi/2$ -carrier rotations on the target qubit this gives a controlled-NOT gate.

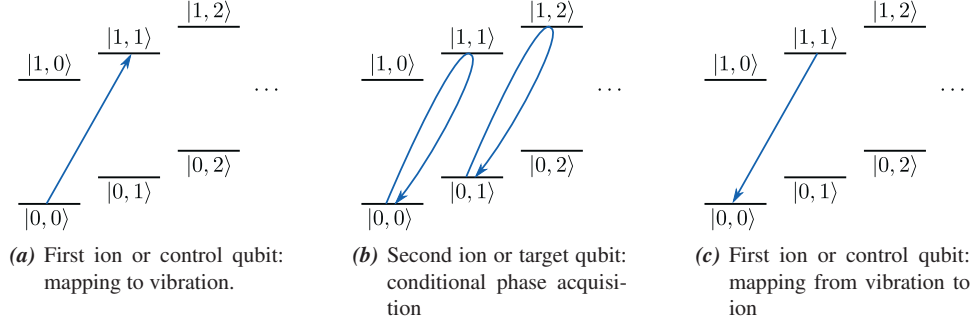
With the notation where the first factor of the tensor product denotes the first ion, the second factor the second ion and the third factor the vibration, eventually, starting with the different basis states, we have done the evolution

$$\begin{aligned}
 |0\rangle |0\rangle |0\rangle &\rightarrow |0\rangle |0\rangle |0\rangle \rightarrow |0\rangle |0\rangle |0\rangle \rightarrow |0\rangle |0\rangle |0\rangle \\
 |0\rangle |1\rangle |0\rangle &\rightarrow |0\rangle |1\rangle |0\rangle \rightarrow |0\rangle |1\rangle |0\rangle \rightarrow |0\rangle |1\rangle |0\rangle \\
 |1\rangle |0\rangle |0\rangle &\rightarrow -i |0\rangle |0\rangle |1\rangle \rightarrow i |0\rangle |0\rangle |1\rangle \rightarrow |1\rangle |0\rangle |0\rangle \\
 |1\rangle |1\rangle |0\rangle &\rightarrow -i |0\rangle |1\rangle |1\rangle \rightarrow -i |0\rangle |1\rangle |1\rangle \rightarrow -|1\rangle |1\rangle |0\rangle.
 \end{aligned} \tag{3.27}$$

That is, if the control qubit is set to 1 the target qubit is subject to a phase flip. To make a controlled-NOT from this controlled phase gate, remember that bit flips can be described as phase flips in the Hadamard rotated basis, and therefore enclosing this sequence by Hadamard gates on the target qubit would give a controlled-NOT gate. Since Hadamard gates are not directly realizable with single pulses in ion trap quantum computation, instead, the controlled phase gate can be enclosed by  $\pi/2$ -carrier pulses around the negative and positive  $y$ -axis, which gives the same result here.<sup>8</sup>

**Composite pulses CNOT gate** The Cirac-Zoller CNOT gate requires an additional auxiliary atomic level. This requirement can be circumvented by using composite pulses, a technique already used in NMR frameworks, now applied to ion trap quantum computation [Childs and Chuang, 2001]. One possible pulse sequence for a controlled phase gate is

<sup>8</sup>It is easy to check that  $CNOT(1, 2) = (\mathbb{1} \otimes R_y(\pi/2))CZ(1, 2)(\mathbb{1} \otimes R_{-y}(\pi/2))$ , with  $CZ$  denoting the controlled phase gate.



**Figure 3.6** – Composite pulses scheme for a controlled phase gate. The internal state of the control ion is mapped to the vibrational quantum state (a), the target qubit acquires a phase flip (b) conditioned on the vibrational mode and the motion is mapped back to the control ion (c). Enclosed between two  $\pi/2$ -carrier rotations on the target qubit this gives a controlled-NOT gate.

given by [Schmidt-Kaler et al., 2003b]. Here blue sideband pulses are used instead of red sideband pulses (see figure 3.6).

1. The first step (figure 3.6(a)) is again the mapping of the internal state of the control ion to the vibrational mode. This time a blue sideband  $\pi$ -pulse is used, that we denote as  $R^{\text{bsb}}(\pi, 0)$ , where the rotation angle is  $\pi$  and the rotation axis is defined by the phase  $\phi = 0$  (see equation (3.26)). Any population in the state  $|0, 0\rangle$  gets transferred to the state  $|1, 1\rangle$ , while any population in the  $|1, 0\rangle$  state does not change.
2. A composite pulse technique is being applied to the target ion in order to do  $2\pi$ -rotations between the states  $|0, 0\rangle$  and  $|1, 1\rangle$  as well as  $|0, 1\rangle$  and  $|1, 2\rangle$  (see figure 3.6(b)). The result is that the states  $|0, 0\rangle$ ,  $|0, 1\rangle$ , and  $|1, 1\rangle$  acquire a (local) phase of  $-1$ , whereas the state  $|1, 0\rangle$  remains unaffected. In order to realize this, the composite pulse technique is needed. Remember that the coupling strength on the sidebands depends on the vibrational quantum state (figure 3.4, equation (3.22)). That is, driving a blue sideband pulse for a specific time or rotation angle on the  $|0, 0\rangle \leftrightarrow |1, 1\rangle$  transition, the rotation angle in the  $|0, 1\rangle \leftrightarrow |1, 2\rangle$  subspace undergoes a rotation that is scaled by a factor of  $\sqrt{2}$ . Thus, to achieve the same rotation on both subspaces a composite pulse is applied, which is described later on in more detail.
3. The motional state is mapped back to the first ion with a  $R^{\text{bsb}}(\pi, \pi)$  pulse.



The development of the basis states goes according to

$$\begin{aligned}
 |0\rangle|0\rangle|0\rangle &\rightarrow -i|1\rangle|0\rangle|1\rangle \rightarrow i|1\rangle|0\rangle|1\rangle \rightarrow -|0\rangle|0\rangle|0\rangle \\
 |0\rangle|1\rangle|0\rangle &\rightarrow -i|1\rangle|1\rangle|1\rangle \rightarrow i|1\rangle|1\rangle|1\rangle \rightarrow -|0\rangle|1\rangle|0\rangle \\
 |1\rangle|0\rangle|0\rangle &\rightarrow |1\rangle|0\rangle|0\rangle \rightarrow -|1\rangle|0\rangle|0\rangle \rightarrow -|1\rangle|0\rangle|0\rangle \\
 |1\rangle|1\rangle|0\rangle &\rightarrow |1\rangle|1\rangle|0\rangle \rightarrow |1\rangle|1\rangle|0\rangle \rightarrow |1\rangle|1\rangle|0\rangle.
 \end{aligned} \tag{3.28}$$

This realizes a controlled phase gate (up to a global phase factor of  $-1$ ). To make this a controlled-NOT gate this sequence can be enclosed by a  $R^{\text{car}}(\frac{\pi}{2}, -\frac{\pi}{2})$  (beforehand) and a  $R^{\text{car}}(\frac{\pi}{2}, \frac{\pi}{2})$  pulse (afterwards).

**Composite pulse phase gate** Here we answer the question how a  $2\pi$ -rotation can be done simultaneously on both subspaces of different phonon numbers, although the coupling strengths differ by a factor of  $\sqrt{2}$ . Instead of a single pulse, a composite pulse is done, which consists of the pulse sequence [Schmidt-Kaler et al., 2003a]

$$\begin{aligned}
 R_{\text{phase}} = R^{\text{bsb}}\left(\pi\sqrt{n+1}, 0\right) R^{\text{bsb}}\left(\pi\sqrt{\frac{n+1}{2}}, \frac{\pi}{2}\right) \\
 R^{\text{bsb}}\left(\pi\sqrt{n+1}, 0\right) R^{\text{bsb}}\left(\pi\sqrt{\frac{n+1}{2}}, \frac{\pi}{2}\right), \tag{3.29}
 \end{aligned}$$

where  $n$  denotes the smaller phonon number of the transition. In the  $|0, 0\rangle \leftrightarrow |1, 1\rangle$  subspace this gives

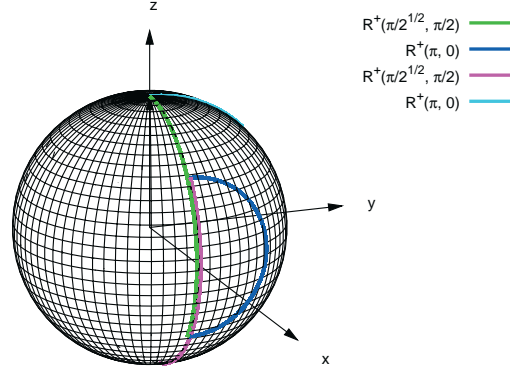
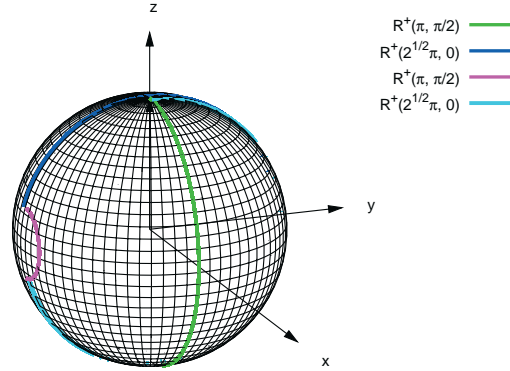
$$R_{\text{phase}} = R^{\text{bsb}}(\pi, 0) R^{\text{bsb}}\left(\frac{\pi}{\sqrt{2}}, \frac{\pi}{2}\right) R^{\text{bsb}}(\pi, 0) R^{\text{bsb}}\left(\frac{\pi}{\sqrt{2}}, \frac{\pi}{2}\right), \tag{3.30}$$

and in the  $|0, 1\rangle \leftrightarrow |1, 2\rangle$  subspace the result is

$$R_{\text{phase}} = R^{\text{bsb}}\left(\pi\sqrt{2}, 0\right) R^{\text{bsb}}\left(\pi, \frac{\pi}{2}\right) R^{\text{bsb}}\left(\pi\sqrt{2}, 0\right) R^{\text{bsb}}\left(\pi, \frac{\pi}{2}\right). \tag{3.31}$$

This sequence does an effective  $2\pi$ -pulse in both subspaces and can best be understood by looking at the trajectories on the Bloch sphere representation of both subspaces (figure 3.7). The involved states follow different paths on the Bloch sphere due to different coupling strengths, but eventually, they all acquire a phase of  $-1$ .

Summarizing the observations from above, we see how a universal set of quantum gates can be realized in ion trap quantum computation. Together with long decoherence times and the possibility to initialize and readout qubits, this shows that ion trap quantum computation fulfills all criteria that are absolutely necessary for a promising quantum computer architecture. The drawback is that it is not scalable to very high numbers of qubits in a single trap,


 (a)  $|0, 0\rangle \leftrightarrow |1, 1\rangle$  subspace

 (b)  $|0, 1\rangle \leftrightarrow |1, 2\rangle$  subspace

**Figure 3.7** – Representation of the composite pulse sequence on the Bloch sphere of the respective subspace. Starting at the top, which represents the state  $|0, 0\rangle$  ( $|0, 1\rangle$ ) a  $\pi/\sqrt{2}$ -rotation ( $\pi$ -rotation) around the  $y$ -axis is performed, followed by a  $\pi$ -rotation ( $\sqrt{2}\pi$ -rotation) around the  $x$ -axis. This sequence is done twice. The result is that the basis states all get a phase shift of  $-1$ . Although the coupling strengths and therefore the angles of rotation in both subspaces differ by a factor of  $\sqrt{2}$ , it is possible to realize an effective  $2\pi$ -rotation in both subspaces simultaneously. Figure modified from [Häffner et al., 2008] with change of rotation axes.

because the approximative description of a linear chain of ions in a harmonic potential will fail (for detail see e.g. [Häffner et al., 2008]). That is why currently much effort is spent in the development of segmented traps and methods for shuttling ions [Häffner et al., 2008]. There are also concepts for traps interconnected by photons [Cirac et al., 1997]. A very interesting concept is to use distributed coupled ion traps, where in each trap a logical qubit is encoded using quantum error correction techniques [Oi et al., 2006], giving a solution to the scalable integration of fault-tolerant quantum error correction.<sup>9</sup> Since trapped ions are very delicate to noise and their handling is experimentally demanding, simulations can be a valuable tool for the improvement of future ion trap quantum computation devices.

**Possible generalizations of the model** This section gives an outlook to possible generalizations of the simulation model. Currently we are considering a one-dimensional system only. We are looking at the axial direction, where the ions are confined by a harmonic potential. Furthermore, our model is limited to one vibrational mode only, i.e. the center-of-mass mode of the string of ions. For more realistic simulations the radial directions should be included as well as additional vibrational modes.

**Three dimensional potential:** As mentioned earlier, the static potential over the end caps of the ion trap creates a harmonic potential in axial direction

$$V_{ax} = \frac{1}{2}m\omega_z^2 z^2, \quad (3.32)$$

with  $z$  being the axial direction. The potential in radial direction can be described as quasi harmonic potential [Leibfried et al., 2003b],

$$V_{rad} = \frac{1}{2}mW(t)x^2, \quad (3.33)$$

with

$$W(t) = \frac{\omega_{rad}^2}{4} [a_x + 2q_x \cos(\omega_{rad}t)]. \quad (3.34)$$

The parameters  $a_x$  and  $q_x$  are geometric device parameters of the individual ion trap and  $\omega_{rad}$  is in the radio frequency regime. Although the Hamiltonian for the radial direction is time-dependent and no stationary states exist, it can be described effectively as a harmonic oscillator. The solution is a secular motion that is superimposed by a micromotion, which can approximatively be neglected, since the operations of interest rely on resonant interaction at the secular frequencies, so it can be averaged over the (much higher frequency) micromotion [Wineland et al., 1998; Leibfried et al., 2003b].

All three dimensions can be treated as a (pseudo-) harmonic potential and therefore the problem is separable into three one-dimensional problems. Treating all three dimensions

---

<sup>9</sup>Still, the phonon bus limits parallel operations within a single trap, which are essential for fault-tolerant quantum error correction (see section 2.3.3.2).

would require replacing  $kx$  by the scalar product  $\vec{k} \cdot \vec{r}$  and the operator  $e^{i\eta(a^\dagger+a)}$  by  $e^{i\vec{k} \cdot \vec{r}} = \prod_m e^{i\eta_m(a_m^\dagger+a_m)}$ , that allows processes that change all oscillators simultaneously.

At the moment our simulator considers a single dimension only. Although it can also deal with the radial directions, the axial direction is actually the direction of interest for quantum computation.

**Multiple ions:** So far, the description has been done for a single ion in a harmonic potential. Going to multiple ions requires some modifications. The part of the Hamiltonian describing the motion and the internal state  $H_0$  becomes

$$H_0 = \sum_j \left( \frac{p_j^2}{2m_j} + \frac{m_j}{2} \omega_t^2 x_j^2 + \frac{1}{2} \hbar \omega_{a,j} \sigma_{z,j} \right), \quad (3.35)$$

with  $j$  denoting the number of the ion. The interaction term  $H_1$  (see equation (3.6)) in the general case becomes [Gulde, 2003]

$$H_1 = \frac{1}{2} \hbar \sum_{j,m} \Omega_j (\sigma_{+,j} + \sigma_{-,j}) \left( e^{i(\pm\eta_{j,m}(a_m^\dagger+a_m)-\omega_t t+\phi_j)} + e^{-i(\pm\eta_{j,m}(a_m^\dagger+a_m)-\omega_t t+\phi_j)} \right), \quad (3.36)$$

where  $j$  labels the ion and  $m$  describes the motional modes taken into account. The Lamb-Dicke factors  $\eta_{j,m}$  are now defined by

$$\eta_{j,m} = k \cos \beta \sqrt{\frac{\hbar}{2M\omega_m}}, \quad (3.37)$$

where  $M$  is the total mass and  $\omega_m$  is the frequency of the respective mode (refer to [James, 1998] for a list of eigenvectors and eigenvalues for normal modes for up to 10 ions). The sign of  $\eta_{j,m}$  depends on the mode as well as on the number  $j$  of the ion (where the eigenvector determines towards which direction the ion is moving).

Since in the experiment only a single ion is addressed at one time, we can drop the sum over  $j$  in our simulations. In experiments usually the lowest energy eigenmode, the collective center of mass motion is used. Therefore, currently, we limit our simulation to this mode only, so that eventually, we end up with equation (3.6). Here we just have a factor of  $N^{-1/2}$  for the Lamb-Dicke parameter and therefore in the occurring Rabi frequencies of the sideband transitions.

### 3.2 Dynamic Quantum Computer Simulator for Ion Traps (DyQCSI)

We are developing a *Dynamic Quantum Computer Simulator for Ion Traps (DyQCSI)* with the goal to support experimental groups working on ion trap quantum computation. Experimentalists have expressed the demand for simulational guidance for their experiments. There is need for a quantification of error influences from different sources [Häffner et al., 2008], because the stable operation of the ion trap quantum computation device is an experimentally highly demanding task. Scalability requires optimal control, i.e., as a prerequisite a good understanding of what effects various error sources are causing, and how large those effects are. Another important task where numerical simulations can be an essential tool, is pulse optimizations, i.e. pulse sequence optimizations as well as pulse shape optimizations.

We implement a numerical model that describes the relevant basic physics of the ion trap quantum computer and we numerically solve the Schrödinger equation (equation (2.9)), that describes the time evolution of the system (equation (2.11)). This simulation package is a starting point and various extensions can be added successively to account for specific physical phenomena.

As described in section 3.1, analytical examinations are always approximations to the real experimental situation, that in some cases are oversimplified. For example, the rotating wave approximation, assuming that only one transition at a time is relevant, is usually not justified<sup>10</sup> [Häffner et al., 2008]. With our numerical integration method we do not apply such approximations and therefore, our simulation handles transitions correctly.

**Simulation Hamiltonian** The Hamiltonian we use for our simulation describes the center of mass motion of a string of  $n$  ions, i.e. two-level quantum systems, in one dimension:

$$H(t) = \underbrace{\frac{p^2}{2M}}_{H_{1,1}} + \underbrace{\frac{M}{2}W(t)x^2}_{H_{1,2}} + \underbrace{\frac{\hbar\omega_a}{2}\sum_{i=1}^n\sigma_z^{(i)}}_{H_2} + \underbrace{\hbar\Omega\sigma_x^{(j)}\cos(kx - \omega_t t + \phi)}_{H_3}. \quad (3.38)$$

We group the Hamiltonian into different parts.  $H_{1,1}$  and  $H_{1,2}$  describe the kinetic and potential term of the harmonic oscillator, with  $p$  being the momentum,  $M$  the total mass of all ions,  $x$  describing the position of the center of mass and  $W(t)$  a (possibly time-dependent) potential. For the axial direction of the trap  $W(t) = W = \omega_t^2$ , i.e., there is only a time-independent harmonic potential. For the radial directions  $W(t)$  is given by equation (3.34). A potential time-dependent error source could also be introduced at this point.  $H_2$  is the spin part describing the internal electronic structure of the  $n$  ions as a

<sup>10</sup>This is because the Lamb-Dicke parameter  $\eta$  (equation (3.5)) is small and the carrier transition is stronger than the sideband transition by a factor of  $1/\eta$ .

two-level quantum system with  $\sigma_z$  being the Pauli-Z matrix.  $H_3$  describes the laser-ion interaction, where a laser with frequency  $\omega_l$ , wavenumber  $k$  and phase  $\phi$  interacts with the  $j^{\text{th}}$  ion.<sup>11</sup>  $\Omega$  is the Rabi frequency, that is determined by the intensity of the laser amongst other things and  $\sigma_x^{(j)}$  is the Pauli-X matrix acting on the  $j^{\text{th}}$  ion.

We make the ansatz to solve the Schrödinger equation in position space. For that, we discretize the one-dimensional space of the trap dimension into  $2^s$  sampling points. The complete state vector of the system is described by  $s2^n$  complex-valued amplitudes. We store the state vector with  $s$  consecutive entries for each spin orientation.

For solving the Schrödinger equation we use the 2<sup>nd</sup> order Suzuki-Trotter formula [De Raedt, 1987], i.e., given the wave function at a time  $t$ , the wave function at the next timestep  $t + \tau$  is determined by

$$|\psi(t + \tau)\rangle = e^{-\frac{i}{\hbar}H_2\frac{\tau}{2}}e^{-\frac{i}{\hbar}H_3\frac{\tau}{2}}e^{-\frac{i}{\hbar}H_{1,2}\frac{\tau}{2}}\underbrace{e^{-\frac{i}{\hbar}H_{1,1}\tau}}_{\text{use FFT}}e^{-\frac{i}{\hbar}H_{1,2}\frac{\tau}{2}}e^{-\frac{i}{\hbar}H_3\frac{\tau}{2}}e^{-\frac{i}{\hbar}H_2\frac{\tau}{2}}|\psi(t)\rangle + \mathcal{O}(\tau^3). \quad (3.39)$$

The overall error after the integration time  $T$  or  $\frac{T}{\tau}$  timesteps is of order  $\mathcal{O}(\tau^2)$ .

For the kinetic term  $H_{1,1}$ , that is put in the middle of the Suzuki-Trotter expansion, a change to the eigenbasis is advisable, so that  $e^{-\frac{i}{\hbar}H_{1,1}\tau}$  becomes a diagonal transformation, i.e.

$$e^{-i\frac{\tau}{2\hbar M}p^2}|\psi\rangle = \sum_k e^{-i\frac{\tau}{2\hbar M}p^2}|k\rangle\langle k|\psi\rangle \quad (3.40)$$

and

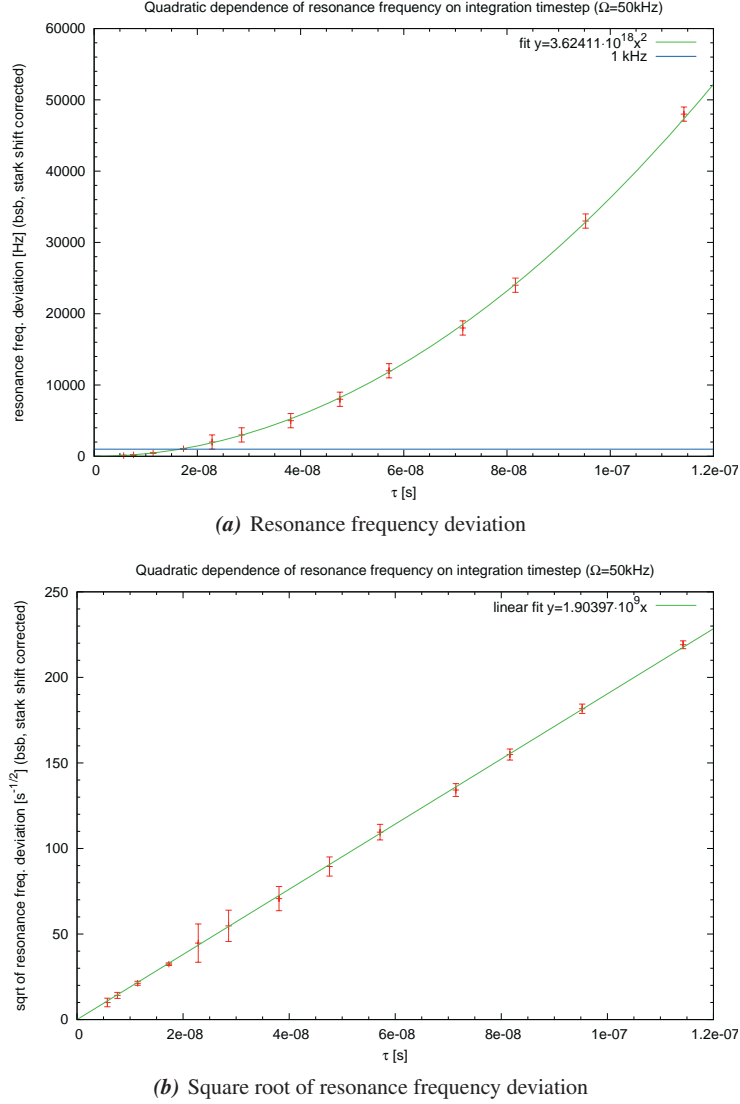
$$p|k\rangle = \hbar k|k\rangle. \quad (3.41)$$

The transformation from the position space to the reciprocal space via fast Fourier transform (FFT), the application of the momentum operator and the back transformation

$$e^{-i\frac{\tau}{2\hbar M}p^2}\psi(x) \equiv \text{FFT}^{-1}(e^{-i\frac{\tau}{2\hbar M}p^2}\text{FFT}(\psi(x))) \quad (3.42)$$

involve only  $\mathcal{O}(2s \log s)$  operations, compared to  $\mathcal{O}(s^2)$  operations for a direct integration of the momentum operator. For the fast Fourier transform we use the FFTW library [Frigo and Johnson, 2005].

The error coming from the Suzuki-Trotter expansion should be  $\mathcal{O}(\tau^2)$ . Figure 3.8 shows the shift of the first blue sideband resonance frequency (see figure 3.11) depending on the



**Figure 3.8** – First blue sideband resonance frequency deviation depending on the integration timestep  $\tau$  using the Suzuki-Trotter decomposition. The Rabi frequency has been set to  $\Omega = 50 \text{ kHz}$ . The second order Suzuki-Trotter decomposition is supposed to exhibit a quadratic error behavior. This error behavior could be confirmed by looking at the resonance frequency. From (a) it can be deduced how small the integration timestep  $\tau$  has to be for a given precision, e.g. 1 kHz. In (b) the square root of the deviation is taken and the quadratic behavior is obvious.

integration timestep  $\tau$ .<sup>12</sup> Within our simulations we make sure that the error coming from the Suzuki-Trotter decomposition is sufficiently small for the analyses that we make.

Apparently, the AC Stark shift  $\delta_{AC}$ , which is given by

$$\delta_{AC} = \frac{1}{2} \left( 1 + \frac{\eta^2}{2} \right) \left( \frac{\Omega}{\omega_t} e^{-\frac{\eta^2}{2}} \right)^2 \omega_t + \frac{1}{8} \left( \frac{\Omega}{\omega_t} e^{-\frac{\eta^2}{2}} \right)^4 \omega_t + \dots, \quad (3.43)$$

has an effect that is non-negligible. Here  $\eta$  denotes the Lamb-Dicke parameter (equation (3.5)),  $\Omega$  the Rabi frequency and  $\omega_t$  the trap frequency. Equation (3.43) is modified from [Steane et al., 2000] to account for non-negligible  $\eta$  according to the Rabi frequencies given in [Leibfried et al., 2003b]. In the Innsbruck ion trap experiment the AC Stark shift is compensated by a far off-resonant laser of the same intensity as the control laser [Häffner et al., 2008]. Our simulation deals with the effects of the AC Stark shift implicitly, since this effect is included in the Hamiltonian (equation (3.38)).

A first verification of the DyQCSI code is performed by looking at simple Rabi oscillations on the carrier transition and detuned from the carrier transition (figure 3.9). As parameters for our simulator we took values that are close to those used in the Innsbruck ion trap experiment [Häffner et al., 2008], i.e. we are using ions of mass  $m = 40 \text{ g/mol} = 6.64 \cdot 10^{-26} \text{ kg}$  ( $^{40}\text{Ca}^+$ ), with an atomic transition frequency of  $\lambda = 729.147 \text{ nm}$ . The axial trap frequency is set to  $\omega_t = 2\pi \cdot 0.9 \text{ MHz}$ . The coupling strength is set to  $\Omega = \pi / (1.3 \cdot 10^{-5}) \text{ Hz}$ . We evaluate our simulation with two ions in the trap with a total mass of  $M = 2m$ . The simulation results are in very good agreement with the theoretical predictions given by equations (3.23) and (3.24). This verifies the correct behavior of the simulation of the internal spin part and the laser-ion interaction.

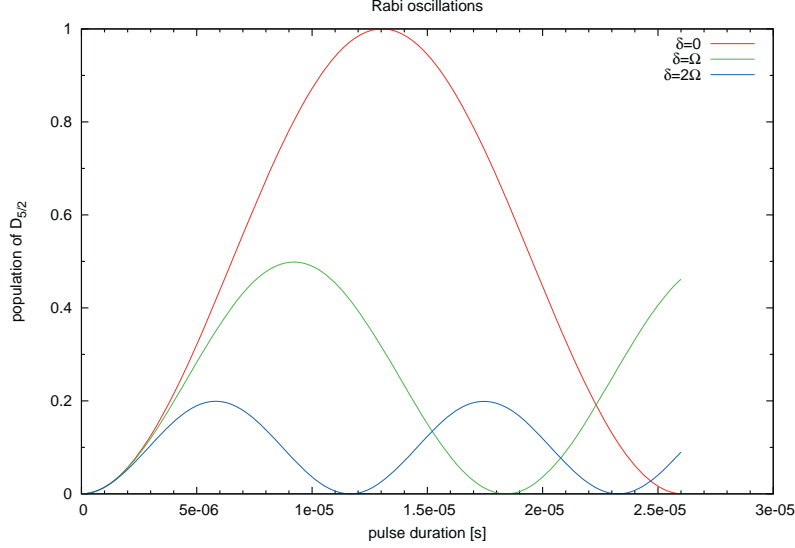
The coupling to the external phonon modes is examined in figure 3.10 and the correct coupling strength on the sideband transition (equation (3.22)) can be confirmed. Being able to do carrier and sideband excitations we are in principle able to do arbitrary quantum operations.

Figure 3.11 shows a frequency scan of the system. The Hamiltonian (equation (3.38)) covers the relevant physics, including carrier transitions, sideband transitions, even of higher orders, as well as Stark shifts of energy levels due to interaction with the laser field. We

<sup>11</sup>The laser is usually incoming under a certain angle  $\theta$  to the motional axis. In fact,  $k$  is the projection of the incoming wave vector  $k'$  of the laser radiation along the motional axis, i.e.  $\vec{k}' \cdot \vec{x} = k'x \cos \theta = kx$ . Otherwise, in a setup as depicted in figure 3.1 a direct addressing of an individual ion along the trap axis would not be possible.

<sup>12</sup>The error behavior in frequency should be equal to that of the energy, provided that  $\omega_a + \omega_t \propto H_{1,1} + H_{1,2} + H_2 + H_3$ . We know the energy eigenvalues of  $H_{1,1} + H_{1,2}$ , which is the spectrum of the harmonic oscillator,  $E_n = \hbar\omega_t (n + \frac{1}{2}) \propto \omega_t$ . The energy eigenvalues of  $H_2$ , that of a spin- $\frac{1}{2}$  system is proportional to  $\omega_a$ . The energy shift from  $H_3$  due to the laser-ion-interaction (AC Stark shift, equation (3.43)) has been corrected for during the determination of the resonance frequencies. In total, the error behavior of the Suzuki-Trotter decomposition ( $\mathcal{O}(\tau^2)$ ) should be seen for the inspection of the resonance frequencies.



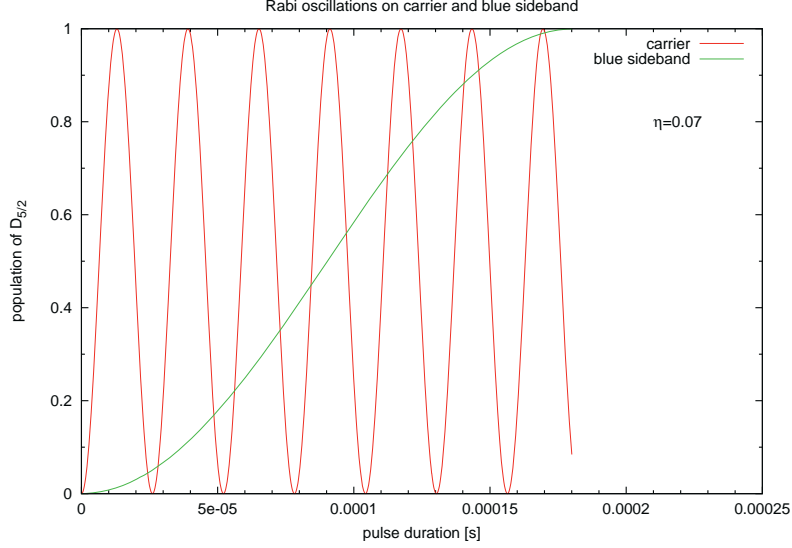


**Figure 3.9** – Rabi oscillations on the carrier transition and detuned from the carrier transition. Here an ion trap quantum computer with two  $^{40}\text{Ca}^+$  ions ( $M = 80 \text{ g/mol}$ ) is driven within a trap with axial frequency  $\omega_t = 2\pi \cdot 0.9 \text{ MHz}$  by a laser with frequency  $\omega_l = 2\pi c / (729.147 \text{ nm})$ ,  $c$  being the speed of light. The resonant carrier transition Rabi frequency is set to  $\Omega = \pi / (1.3 \cdot 10^{-5}) \text{ Hz}$ . The laser interacts with one of the ions and if the laser is exactly on resonance, we see regular Rabi oscillations between the two states of the two-level system. If the laser is off-resonant, i.e. detuned by  $\delta$ , the frequency and amplitudes of the oscillation change according to equations (3.23) and (3.24).

make sure that our simulator covers the basic physics of a system of two-level ions in a harmonic potential interacting with monochromatic laser light. It does not need to make the rotating wave approximation and does not rely on the Lamb-Dicke approximation. Therefore, our simulation handles off-resonant excitations correctly and is exact except for the Suzuki-Trotter expansion, but for this we can quantify the error (figure 3.8).

The geometry of the ion trap and the laser is accounted for by modifying the Lamb-Dicke parameter (equation (3.5)). Figure 3.12 shows a laser that is tuned to the blue sideband transition turned onto the ions under various incident angles to the trap axis. Apparently, incoming radiation orthogonal to the trap axis is not able to excite any phonon mode. Radiation along the trap axis is only theoretically possible for more than a single ion in the trap. Experiments usually use a setup that ensures a small Lamb-Dicke factor  $\eta$ . A small Lamb-Dicke factor will lead to long sideband pulses limiting the speed of operations, but there are experimental constraints that require a small Lamb-Dicke factor (such as the sideband cooling of the ions [Häffner et al., 2008]).

We now demonstrate how DyQCSI can be used to simulate the evolution of a controlled-

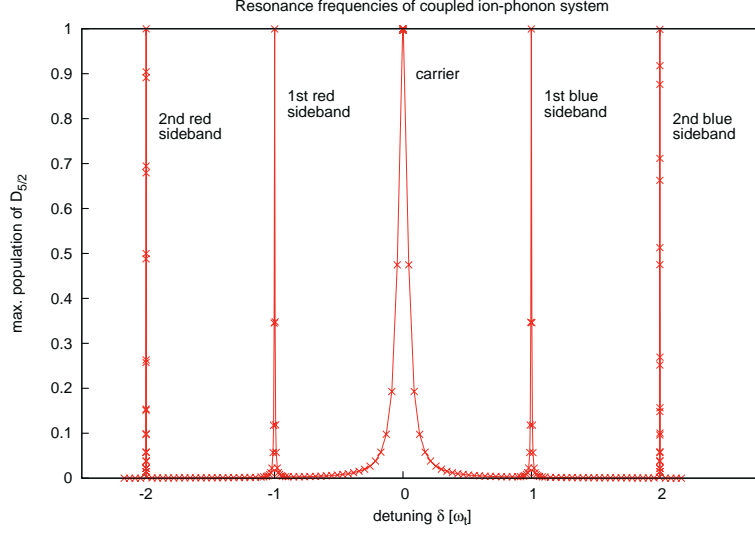


**Figure 3.10** – Rabi oscillation on the carrier transition and on the first blue sideband. The same parameters are used as in figure 3.9. This results in a Lamb-Dicke parameter  $\eta = 0.07$ . The coupling strength of the blue sideband transition is a factor of  $\eta$  weaker than the carrier transition (equation (3.22)). Thus, pulses on the sidebands take much more time than carrier pulses and are a limiting factor for the speed of ion trap quantum computation devices.

NOT gate. The simulation approach gives us the advantage, that we can access all parameters during the execution of any algorithm. For example, figure 3.13 shows the development of the  $D_{5/2}$  population during a CNOT gate operation. This plot is the result of a single simulation run. A similar plot with experimental results can be found in [Schmidt-Kaler et al., 2003b].<sup>13</sup> Note that for the experimental approach, each point is the result of 100 experimental cycles [Schmidt-Kaler et al., 2003b] and for the observation of the time evolution hundreds of points have to be measured. With figure 3.13 we have verified that our simulator is capable of describing the dynamics of a controlled-NOT operation.

However, it should be pointed out that the given pulse sequences (see e.g. figure 3.13(a)) are derived from the simplified analytical expressions (equations (3.7)–(3.15)). Since the simulation does not rely on these approximations, but uses the exact Hamiltonian (equation (3.1)), a deviation from the idealized case can be expected. Actually, we see a difference between the predictions of the simplified analytical expressions and our simulation results. In particular, when doing a pulse sequence with changes from carrier excitation to sideband excitation or vice versa, an additional phase shift manifests. But since we have

<sup>13</sup>[Schmidt-Kaler et al., 2003b] use the inverted sequence, so the time evolution of the population is inverted in time.



**Figure 3.11** – Resonance frequencies of the coupled system of ions and phonons. Shown here is the maximum probability of finding the addressed ion in the excited  $D_{5/2}$  state when starting from the  $S_{1/2}$  ground state. The center peak is the carrier transition while the outer peaks are sideband resonances. The resonances are found at detunings of multiple integers of the trap frequency  $\omega_t$ , i.e. at frequencies inducing phononic (de-)excitations additional to atomic transitions. The second order sideband transitions have very sharp resonance frequencies, because the coupling is very weak (equation (3.19)).

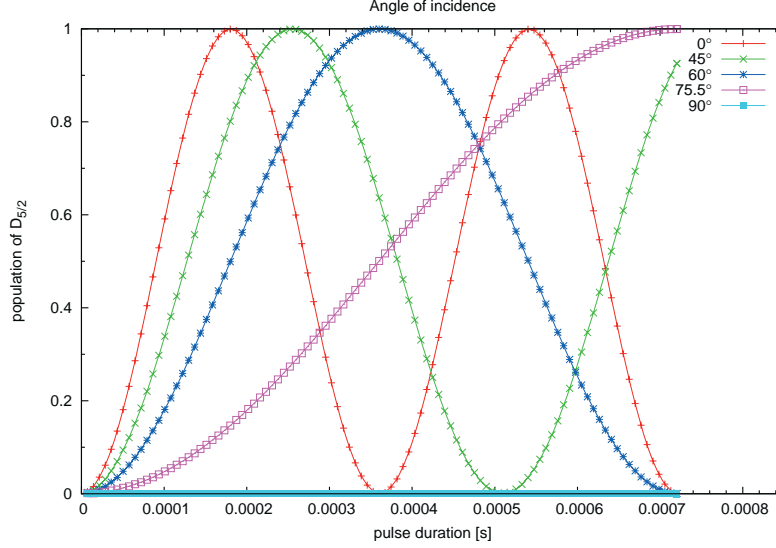
full information about the state of our system, as we describe in the following, we used that information to compensate for the deviation. Keep in mind that for the description of the controlled-NOT gate in this section, this adjustment is done implicitly. A more detailed analysis of these probably non-negligible effects will be done in the future.

With our simulator we are not confined to the observation of the  $D_{5/2}$  population, as in experiment, but we can also have a look at the position space wave function. We will show this for the controlled phase gate. The eigenstates of the quantum harmonic oscillator in position space are given by [Cohen-Tannoudji et al., 1978]

$$\langle x | \psi_n \rangle = \psi_n(x) = \sqrt{\frac{1}{2^n n!}} \left( \frac{M\omega_t}{\pi\hbar} \right)^{\frac{1}{4}} e^{-\frac{M\omega_t}{2\hbar} x^2} H_n \left( \sqrt{\frac{M\omega_t}{\hbar}} x \right), \quad (3.44)$$

with total mass  $M$  and circular frequency  $\omega_t$ . The Hermite polynomials  $H_n$  are defined as

$$H_n(x) = (-1)^n e^{x^2} \frac{d^n}{dx^n} e^{-x^2}. \quad (3.45)$$



**Figure 3.12** – Blue sideband pulse for various incident angles of laser radiation. The coupling strength and thus the speed of Rabi oscillations depend on the angle of the incoming laser radiation. Laser light orthogonal to the trap axis is not able to excite axial phonon modes. The coupling becomes stronger the more the incident angle gets closer to the trap axis.

The coherent ground state of the harmonic oscillator is a Gaussian

$$\langle x|\psi_0\rangle = \psi_0(x) = \left(\frac{M\omega_t}{\pi\hbar}\right)^{\frac{1}{4}} e^{-\frac{M\omega_t}{2\hbar}x^2}. \quad (3.46)$$

The first and second excited states are given by

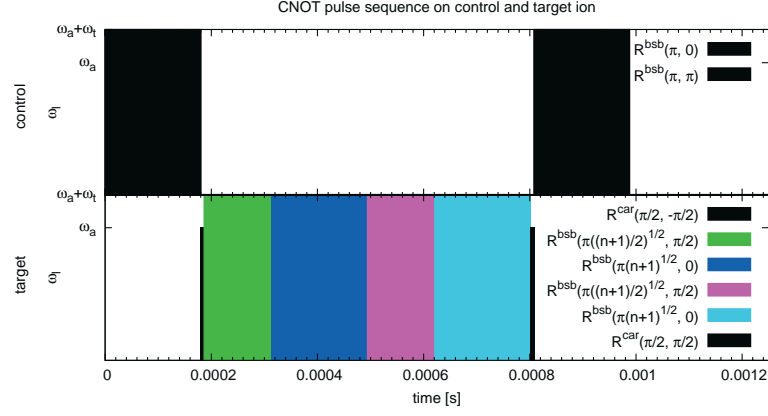
$$\langle x|\psi_1\rangle = \psi_1(x) = \left(\frac{4}{\pi}\left(\frac{M\omega_t}{\hbar}\right)^3\right)^{\frac{1}{4}} x e^{-\frac{M\omega_t}{2\hbar}x^2} \quad (3.47)$$

and

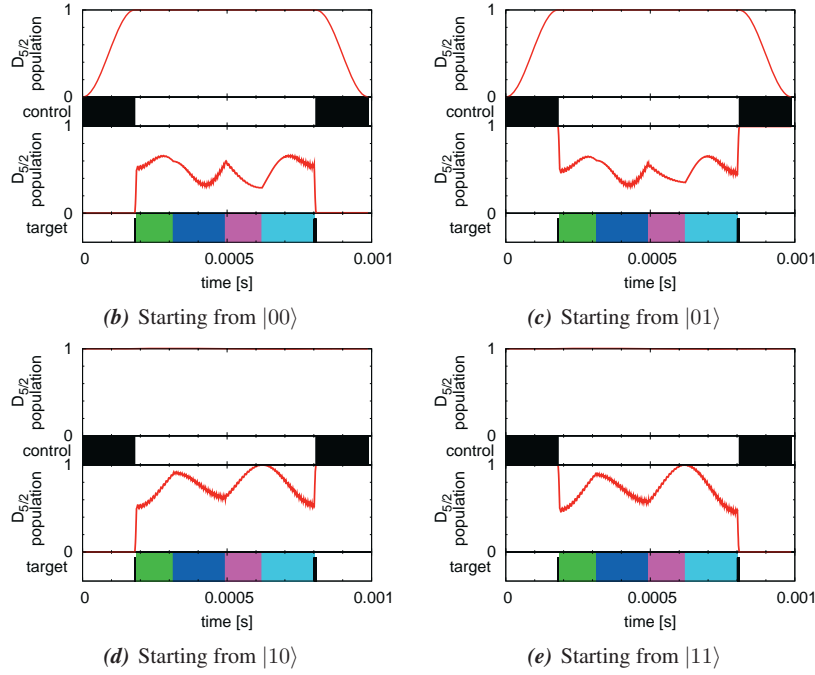
$$\langle x|\psi_2\rangle = \psi_2(x) = \left(\frac{M\omega_t}{4\pi\hbar}\right)^{\frac{1}{4}} \left(2\frac{M\omega_t}{\hbar}x^2 - 1\right) e^{-\frac{M\omega_t}{2\hbar}x^2}. \quad (3.48)$$

For the visualization we look at all possible spin orientations of a two qubit system (figure 3.14). For each spin orientation the spatial probability distribution is examined separately. In these plots one can follow the time evolution of the wave function in position space, e.g., one can see the transformation of  $\psi_0(x)$  to  $\psi_1(x)$  and superpositions of those states. We give an example for both the control and the target qubit starting in the  $|0\rangle$  state.<sup>14</sup>

<sup>14</sup>We use the notation  $|0\rangle \equiv |\uparrow\rangle$  and  $|1\rangle \equiv |\downarrow\rangle$ .



(a) Pulse sequence for CNOT gate (for details see section 3.1) that is used for the examination of the evolution of the  $D_{5/2}$  population in figures (b) – (e).



**Figure 3.13** – Time evolution of the  $D_{5/2}$  population during a controlled-NOT gate. (b) – (e) show all combinations of possible input basis states. Apparently this realizes a CNOT gate. Our simulation gives a complete description of the state vector throughout the sequence. By contrast, in a real experiment, each single data point can only be obtained by numerous experimental runs.

Figure 3.14 shows snapshots at the end of each pulse only. The simulation is in principle able to output the state of the system at intermediate timesteps. The time evolution of the controlled phase gate for other input basis states is available, but omitted here for the sake of brevity.

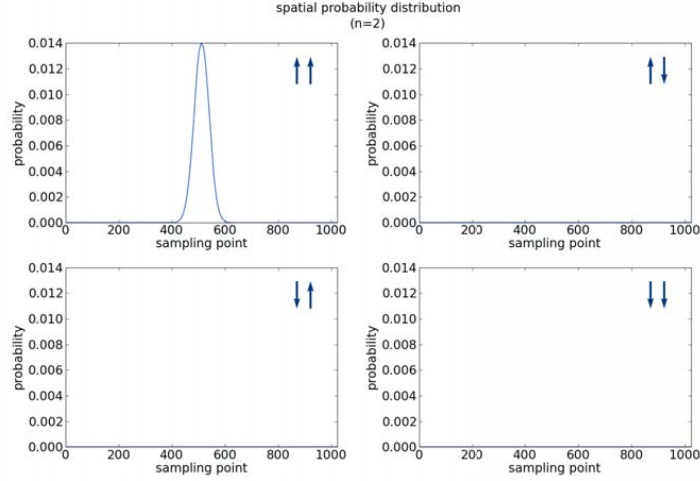
Another possible way to illustrate the time evolution of the ion trap system is to look at the expectation values of the state vector in the Bloch sphere representation (figure 3.15). Here the Bloch spheres represent the subspaces  $|0,0\rangle \leftrightarrow |1,0\rangle$ ,  $|0,1\rangle \leftrightarrow |1,1\rangle$ ,  $|0,0\rangle \leftrightarrow |1,1\rangle$ , and  $|0,1\rangle \leftrightarrow |1,2\rangle$ . Since we examine a two-qubit system with control and target qubit, the state at each timestep can be described by eight Bloch spheres. Analogous to figure 3.14 we start with both qubits initialized to  $|0\rangle$  and no phonon excitation and show snapshots at the end of each pulse of the controlled phase gate. With our simulator we are able to give the time evolution of the expectation values for all three axes. This sort of visualization will be useful for the analysis of future simulation runs.

Additionally, our simulator offers the possibility to work with (squeezed) coherent states resembling the classical motion of a harmonic oscillator. Our simulator can be initialized with any arbitrary states, not necessarily energy eigenstates of the harmonic oscillator. There are several proposals for doing ion trap quantum computation with coherent states, e.g. by [Monroe et al., 1996; Munro et al., 2000]. Our simulator is also suitable for the analysis of these states.

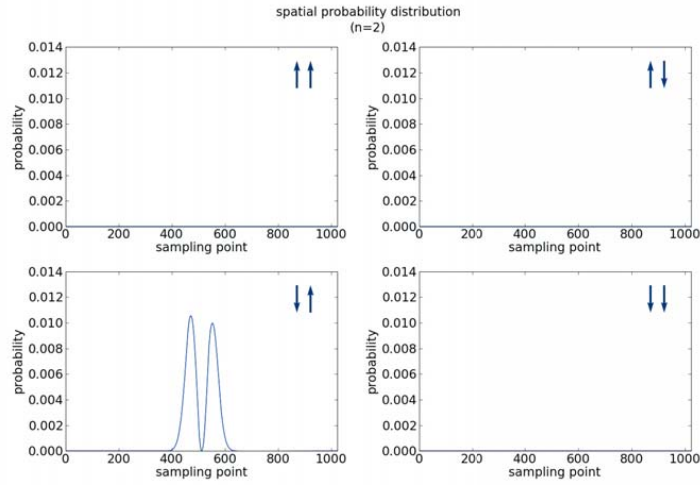
Summarizing the work that has been done so far, we have developed and verified the *Dynamic Quantum Computer Simulator for Ion Traps (DyQCSI)*, that uses a microscopic Hamiltonian that captures the relevant dynamics of an ion trap quantum computation device. Starting from first principles we are able to quantify the effects coming from physics rather than approximations or experimental error sources. We have shown preliminary results for the functionality of the controlled-NOT gate and visualized the time evolution in the Bloch sphere representation as well as in position space. Since DyQCSI works on parallel machines the number of qubits considered in the simulation (up to 16 qubits on JUGENE) can be significantly higher than the number of qubits in today's state-of-the-art quantum computation devices (8 qubits in Innsbruck).

**Outlook** To make DyQCSI a valuable tool for experimentalists, the simulator has to include experimental error sources. The most relevant ones according to [Häffner et al., 2008] are

- AC-stark effect: This effect is already included in the Hamiltonian (equation (3.38)).
- Off-resonant excitations: These are also included in equation (3.38).
- Laser frequency noise: Fluctuations of the laser frequency can be directly included into the simulation. At the moment we drive the simulation with monochromatic laser light at exact carrier and sideband transition frequencies. This can be extended to other frequencies and the laser can be tuned to arbitrary frequency fluctuation behavior.

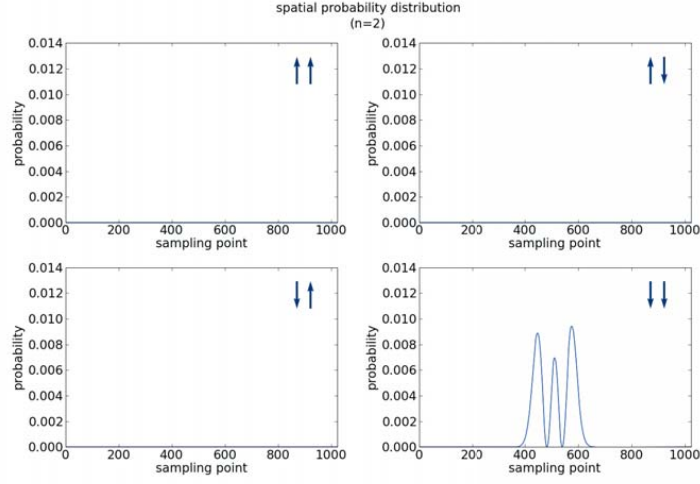


(a) Initial state

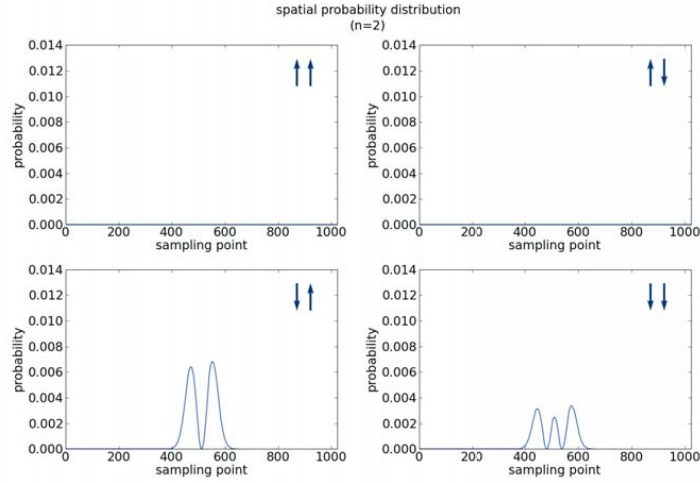


(b) State of control qubit mapped to phonon mode

**Figure 3.14** – Spatial probability distribution during a controlled phase gate. Each figure is divided into four quadrants according to the possible spin orientations. Each quadrant shows the part of the spatial probability distribution for the respective spin orientation. (a) The ground state wave function is given by equation (3.46). (b) After a  $\pi$ -pulse on the blue sideband we see how the population is mapped to the  $|1, 1\rangle$  state. We see that the spin of the control qubit is flipped while a phonon excitation has been created. The probability density is that of the first excited state of the harmonic oscillator (equation (3.47)).



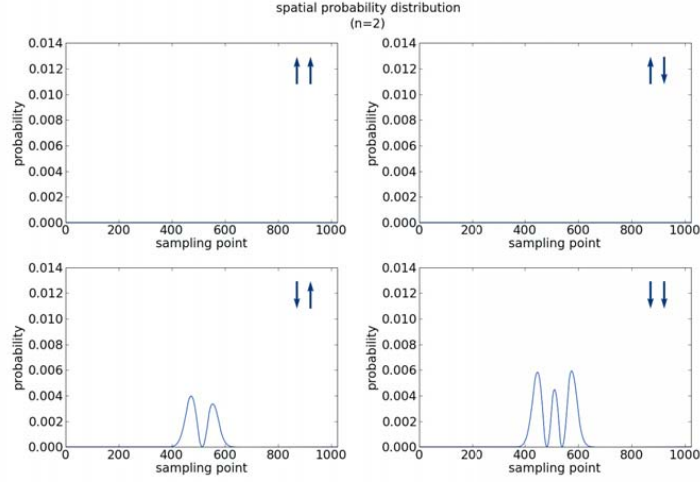
(c) First composite pulse



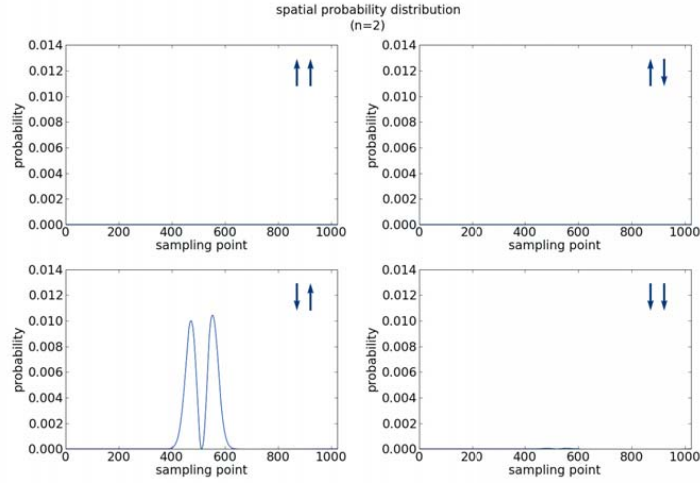
(d) Second composite pulse

**Figure 3.14** – (c) After the first pulse of the composite pulse sequence the population in  $|0, 1\rangle$  is transferred to  $|1, 2\rangle$ , i.e. a state with two phonons (equation (3.48)). (d) The second pulse leaves the state in a superposition of  $|0, 1\rangle$  and  $|1, 2\rangle$ .



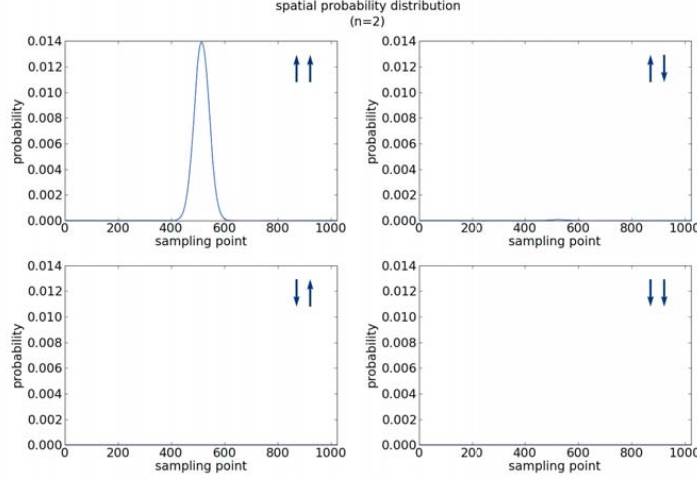


(e) Third composite pulse



(f) Fourth composite pulse

**Figure 3.14** – (e) The third pulse inverts the population of  $|0, 1\rangle$  and  $|1, 2\rangle$ . (f) The fourth pulse completes the composite pulse sequence. The state is again in  $|0, 1\rangle$  and has acquired a phase of  $-1$  (which cannot be seen in the spatial probability distribution).

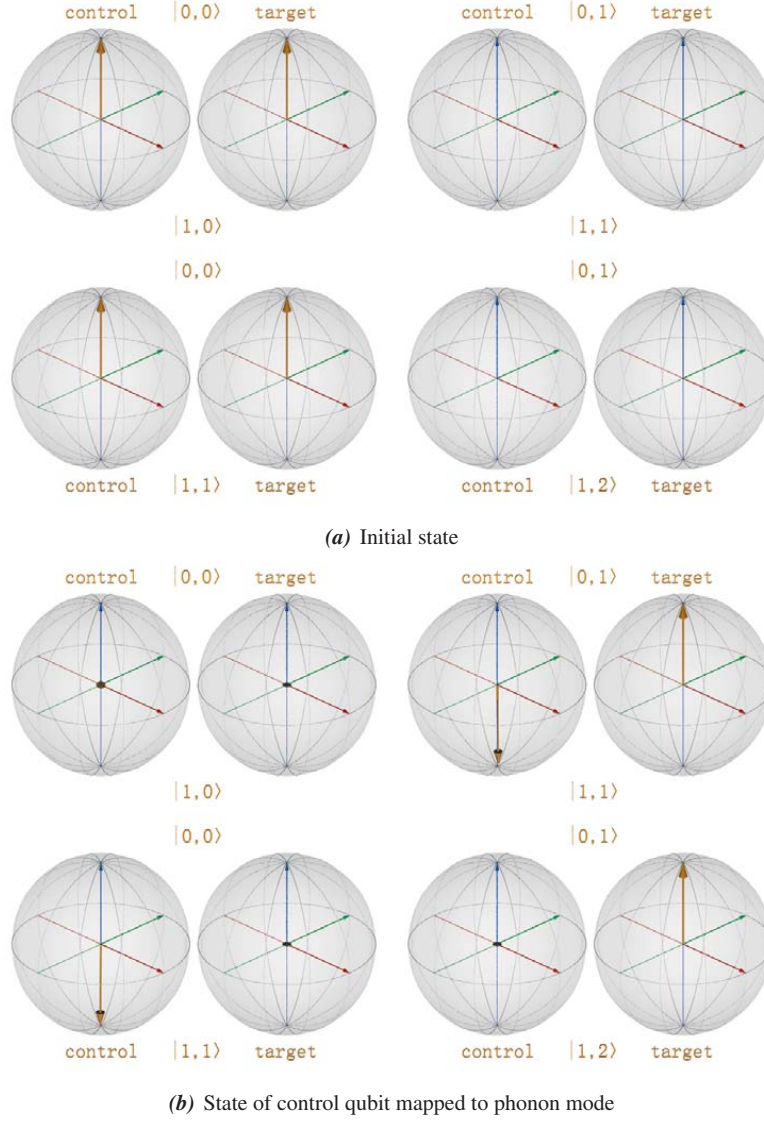


(g) State of phonon mode is mapped back to control qubit

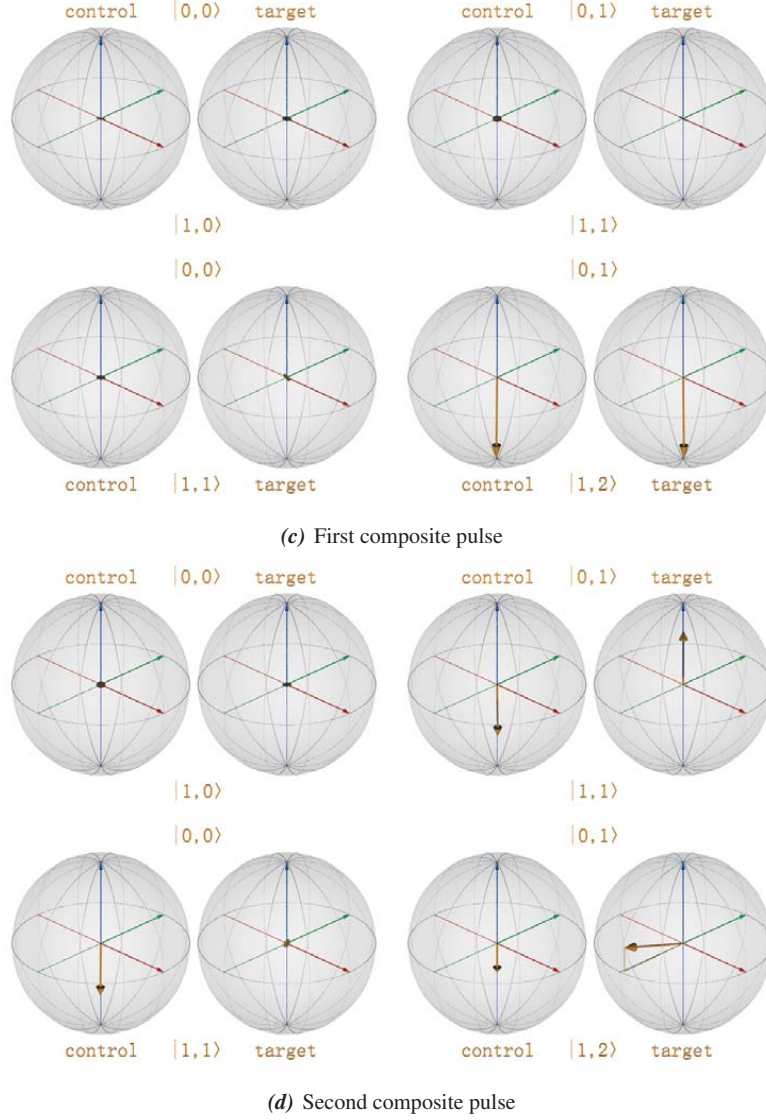
**Figure 3.14 – (g)** The last pulse maps the state of the phonon bus back to the control qubit. We see how the simulation gives us the possibility to follow the time evolution of the wave function in position space, that would not be possible in the experiment. This information might be exploited to improve the performance of ion trap quantum computation devices in the lab.

- **Laser intensity fluctuations:** Since the Rabi frequencies depend on the intensity of the laser, fluctuations of the laser intensity will result in pulse length errors. In our simulation, this can be easily taken into account by introducing a time dependent effective Rabi frequency and parameterizing the occurring fluctuations.
- **Magnetic field noise:** Fluctuations of the magnetic field are a major source of decoherence. They will lead to a fluctuating atomic resonance frequency and can therefore be modeled by a variation of the effective Rabi frequency.
- **Addressing errors:** While addressing a specific ion, residual light can affect other ions. This effect can be modeled by a variation of the effective Rabi frequencies. However, we have to extend the laser-ion interaction term of the Hamiltonian (equation (3.38)) to a sum over all ions and consider complex Rabi frequencies for the ions not targeted, because the phase of the laser light on those will be different.
- **Imperfect ground state cooling:** An averaging with different initial states can account for this.

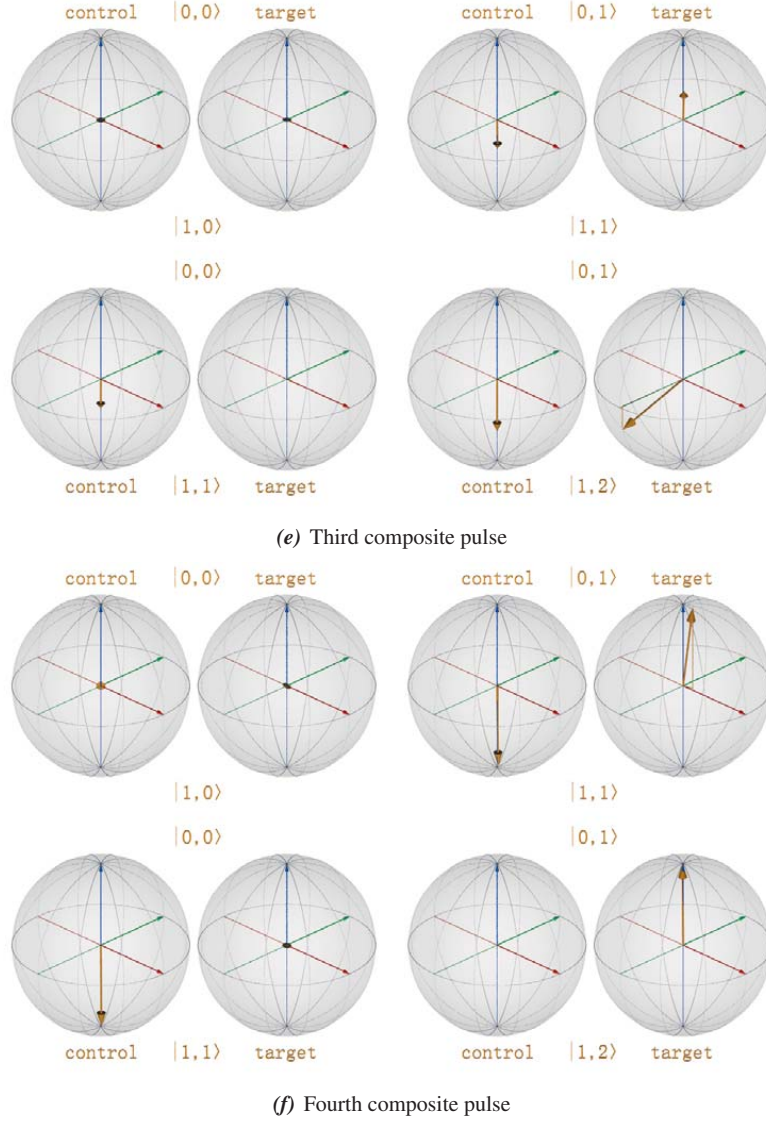
It is possible to implement the error sources into the microscopic Hamiltonian of our simulator and gradually include more and more effects. The model can be extended to include more phonon modes and with the extension to two and three dimensions the simulator can



**Figure 3.15** – Each figure shows the Hilbert subspaces  $|0,0\rangle \leftrightarrow |1,0\rangle$ ,  $|0,1\rangle \leftrightarrow |1,1\rangle$ ,  $|0,0\rangle \leftrightarrow |1,1\rangle$ , and  $|0,1\rangle \leftrightarrow |1,2\rangle$  for each the control and the target qubit after each pulse of the composite pulse controlled phase gate. (a) The initial state of both qubits is  $|0\rangle$ . (b) After a  $\pi$ -pulse on the control qubit on the blue sideband we see how the population is mapped from  $|0,0\rangle$  to the  $|1,1\rangle$  state. The target qubit keeps its internal state, but is now represented in the Bloch sphere representing one phononic excitation.



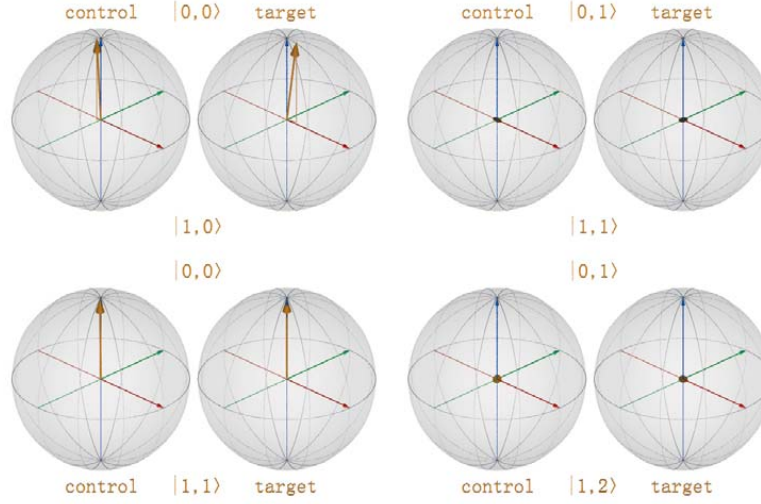
**Figure 3.15** – (c) After the first pulse of the composite pulse sequence the control and the target qubit are in the state  $|1\rangle$ , while the phonon bus is in the state  $|2\rangle$  (compare figure 3.14(c)). (d) The second pulse leaves the target qubit in a superposition of  $|0,1\rangle$  and  $|1,2\rangle$  (compare figures 3.14(d) and 3.7). The control qubit stays in the state  $|1\rangle$ , but the phonon number changes.



**Figure 3.15** – (e) The third and fourth pulse have been depicted theoretically in figure 3.7.

account for radial phonon modes which are a main source of decoherence in the current experimental setup [Häffner et al., 2008].

We intend to implement the error sources into the simulator, so that it can be used to quantify error influences and to examine interconnections between error sources. We want to



(g) State of phonon mode is mapped back to control qubit

**Figure 3.15 – (g)** The last pulse maps the state of the phonon bus back to the control qubit. As expected, we see the same population as in the initial state. The acquired phase of  $-1$  can not be seen in this representation. Minimal deviations are related to phase imprecisions and therefore imprecisions in the choice of rotation axes.

have a tool, that can be used to minimize the sensitivity to control parameters. The advantage of a simulator becomes obvious when it comes to finding optimal design parameters, which would be difficult to determine through countless experimental runs. While in experiment each point of a curve showing the time evolution is a result of numerous (100–1000) experimental runs [Schmidt-Kaler et al., 2003b; Häffner et al., 2008] and gaining a curve with hundreds of points is experimentally very expensive, a single simulation run will immediately give all points of the curve. This will be very helpful for doing parameter studies like pulse shape optimizations, where a single run of the simulator can reveal the effect of a designed pulse shape, whereas in the lab the cost to try various parameters would be extremely high. Besides pulse shape optimizations the simulator can be a useful tool for evaluating designed pulse sequences (e.g. [Khaneja et al., 2005]).





## Chapter 4

### Summary and Outlook

In this thesis the power of error-prone quantum computation devices has been assessed. The instruments for performing these analyses have been developed and verified during the work on this thesis. Two software packages have been created: The *Juelich Massively Parallel Ideal Quantum Computer Simulator (JUMPIQCS)* and the *Dynamic Quantum Computer Simulator for Ion Traps (DyQCSI)*.

JUMPIQCS simulates the evolution of a generic universal quantum computer using the circuit model of quantum computation. Using a complete description of the quantum mechanical system requires exponential resources. Thus, a parallel version for large-scale supercomputers has been developed and optimized for performance and scalability. JUMPIQCS has been extended with an error model for decoherence errors as well as operational imprecisions. This has been used to analyze the sensitivity and robustness to noise of two important quantum algorithms, the quantum Fourier transform, which is the quantum kernel of Shor's algorithm for prime factorization, and Grover's search algorithm. The result from these analyses is that a universal statement about the performance of a quantum computation device in a noisy environment cannot be given, as it depends on the specific quantum algorithm. For the cases, that we have studied, numerical simulations showed that the quantum Fourier transform is more robust to decoherence and operational errors than Grover's algorithm. We relate this to the regular structure of the quantum Fourier transform circuit. The quantum Fourier transform exhibits a threshold like behavior for operational imprecisions, that shows no significant dependency on the system size. Concerning decoherence errors, the numerical results show that the quantum Fourier transform requires a decrement of about one order of magnitude in the single gate error probability when doubling the system size to keep a constant total error level. Grover's algorithm is more susceptible to noise compared to the quantum Fourier transform for similar system sizes. Since no system size independent threshold for operational imprecisions can be observed, the conclusion is that going to larger system sizes proves difficult for Grover's search algorithm unless extraordinarily good control of operational errors is possible. We show how the performance loss



of Grover’s algorithm due to noise can be attenuated by adjusting the number of Grover iterations.

After quantifying the effects of errors on quantum computers, we have analyzed how errors can be actively corrected. We have implemented various quantum error correction techniques (5-, 7-, and 9-qubit scheme) into JUMPIQCS to simulate the effect of quantum error correction circuits. While perfect error-free correction circuits are able to protect quantum information from degrading, this case is unrealistic, because in reality, the correction circuits used are prone to error themselves. Our simulations confirm that using error-prone correction circuits with standard quantum error correction techniques will always fail, i.e., they will introduce more errors than the scheme can correct.

Fault-tolerant quantum error correction techniques can overcome this problem. However, this is not unconditionally true, but there is a necessary condition for the single qubit error rate to be below a certain threshold for fault-tolerant quantum error correction. We incorporated fault-tolerant quantum error correction techniques using Steane’s 7-qubit quantum error correction code into JUMPIQCS and used our simulator to determine this threshold numerically. Our approach does make realistic assumptions about the number of available qubits and uses explicit constructions for the correction circuits.

Studying the case of a sequence of logical Hadamard gates, we obtained numerical results for the thresholds. Using the depolarizing channel as the error model, we found a numerical threshold for fault-tolerant quantum error correction of  $p_{\text{thr}} = (5.2 \pm 0.2) \cdot 10^{-6}$ . If the single qubit error rate is below this threshold, fault-tolerant quantum error correction guarantees an improvement compared to the unprotected case. The other important source of error is operational imprecision. We have numerically determined a threshold for the allowed operational imprecision for fault-tolerant quantum error correction to be functional. For Gaussian distributed operational over-rotations the threshold lies at a standard deviation of  $\sigma_{\text{thr}} = 0.0431 \pm 0.0002$ . This is a rather large value, so that we can conclude, that quantum error correction is especially well suited for the correction of operational imprecisions. In case of systematic over-rotations we have verified that fault-tolerant quantum error correction can always improve the performance of an error-prone device, independent of any threshold.

An analysis of Grover’s search algorithm with only partially protected qubits indicates that a partial encoding of the quantum system is not suitable for the protection of the whole system. Errors from unprotected qubits will propagate to many qubits and thus render quantum error correction useless.

By looking at the example of quantum teleportation we show how entanglement between encoded qubits can be preserved. Additionally, this example reveals that the benefit from quantum error correction will be large enough to justify the cost, only for long algorithms or long storage times of quantum memory.

With JUMPIQCS we have a tool ready, that can be used for the analysis of quantum error correction. It allows to examine under which conditions quantum error correction can

---

give a benefit and to find optimal working points for quantum error correction for arbitrary quantum algorithms. Since future quantum computing devices will most probably incorporate some sort of quantum error correction, JUMPIQCS can also be used as a tool for the assessment of future quantum computing architectures.

A possible extension of JUMPIQCS will be the inclusion of a fully fault-tolerant  $\pi/8$ -gate, which would enable universal quantum computation in a fault-tolerant manner. The fully encoded quantum Fourier transform has not yet been analyzed because of the missing fault-tolerant implementation of the  $\pi/8$ -gate. It would be worth to follow approximative approaches that truncate the precision of required rotations. There are indications that under the presence of noise a limitation of rotation angles does not decrease the performance of the quantum Fourier transform [Barenco et al., 1996]. Yet, it would be interesting to see how quantum error correction performs in this situation. There is also an analysis showing the “scalability of Shor’s algorithm with a limited set of rotation gates” [Fowler and Hollenberg, 2004]. This will help with the analysis of a fully encoded quantum Fourier transform.

While JUMPIQCS simulates a quantum computation device on a generic level, the other simulation code that has been created during the work on this thesis, DyQCSI, deals with the simulation of specific ion trap quantum computation devices. At the moment DyQCSI covers all basic physics of an ion trap quantum computer. It starts from a microscopic Hamiltonian and does not rely on approximations that are usually necessary for an analytical approach. We have demonstrated the basic functionality of DyQCSI by simulating a controlled-NOT gate and we have shown several ways to visualize the state of the system and its time evolution. The access to the complete state of the system during time evolution gives a clear benefit when doing parameter optimizations. DyQCSI is still in development and will be extended to include experimental error sources. We intend to use it as a tool for finding optimal design parameters for ion trap quantum computation devices in the lab. We expect pulse shape and pulse sequence optimizations to be the main application fields for DyQCSI.



# Acknowledgements

Life is complex – it has both  
real and imaginary parts.

*(Anonymous)*

First of all, I want to thank my advisor Professor Thomas Lippert for giving me the opportunity to do this Ph.D. work at the Jülich Supercomputing Centre (JSC). I am grateful for the trust and confidence he has placed in me and for his constant professional support. I also want to thank Dr. Guido Arnold and Dr. Marcus Richter for the friendly acceptance into the quantum information processing group at the JSC, for interesting discussions and last but not least for supervising my work. I would also like to thank Professor Hans De Raedt and Professor Kristel Michielsen for enlightening discussions and for bringing in new ideas. Thanks go to my office mate Sebastian Reuschel for fruitful discussions, proof-reading and help with this thesis. I am much obliged to the JSC operations team, which has done an excellent job providing reliable supercomputing resources. I thank my colleagues at the JSC for creating a pleasant working atmosphere. I am especially thankful for the encouragement from my (former) fellow Ph.D. students, including Dr. Tatjana Streit, Professor Matthias Bolten, Dr. Stefan Krieg, Dr. Tom Schröder, Daniel Becker, Ivo Kabadshow, Robert Speck, and Benjamin Berberich. I would also like to thank the German Research School for Simulation Sciences, that has partly funded my work.



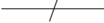
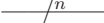
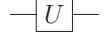


# Appendix A






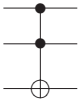


## Quantum Circuit Symbols

The graphical representation of a quantum circuit describing the time evolution of a quantum system usually uses a certain set of symbols summarized in this section. The circuit representation is read from left to right indicating the time evolution. The vertical dimension specifies the size of the system, i.e. the number of qubits.

The following table gives an overview of the graphical symbols used in this thesis. If appropriate, the third column gives the matrix representation of the unitary transform acting on the corresponding subspace of the system.

symbol	description	matrix
	wire representing a single qubit (time goes from left to right)	$\mathbb{1} = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$
	double wire representing a classical bit	1
	wire summarizing multiple qubits	$\mathbb{1} \otimes \dots \otimes \mathbb{1}$
	wire summarizing $n$ qubits	$\underbrace{\mathbb{1} \otimes \dots \otimes \mathbb{1}}_{n \text{ times}}$
	single qubit gate doing a unitary transformation $U$	$U = \begin{pmatrix} u_{11} & u_{12} \\ u_{21} & u_{22} \end{pmatrix}$

## APPENDIX A. QUANTUM CIRCUIT SYMBOLS

symbol	description	matrix
	controlled- $U$ : controlled unitary operation; vertical wires connect control and target qubits; the bullet indicates the controlling qubit; a filled bullet represents a control-on-one, i.e., the operation on the target qubit is done if the control qubit is set to one	$\begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & u_{11} & u_{12} \\ 0 & 0 & u_{21} & u_{22} \end{pmatrix}$
	controlled-NOT (CNOT), which is a controlled- $U$ with $U = X$ being the Pauli- $X$ matrix describing a bit-flip operation (see figure 2.2)	$\begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix}$
	control-on-zero: an open bullet indicates a controlled operation as well, but the operation on the target qubit is done if the control qubit is set to zero	$\begin{pmatrix} u_{11} & u_{12} & 0 & 0 \\ u_{21} & u_{22} & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$
	zero-controlled-NOT: controlled-NOT with control-on-zero	$\begin{pmatrix} 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$
	swap gate: swaps two qubits with each other (see figure 2.26 for a decomposition into a sequence of three CNOT gates)	$\begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$
	Toffoli gate: this 3-qubit operation flips the target qubit if both control qubits are set to one	$\begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix}$
	$C^n$ NOT gate: a generalization of the CNOT ( $=C^1$ NOT) and Toffoli ( $=C^2$ NOT) gate, that flips the target qubit if and only if all control qubits are set to one	$\begin{pmatrix} 1 & & & & & & 0 \\ & 1 & & & & & \\ & & \ddots & & & & \\ & & & 1 & & & \\ 0 & & & & 1 & & \\ & & & & & 0 & 1 \\ & & & & & 1 & 0 \end{pmatrix}$
	projective measurement: the result of this non-unitary transformation is a classical bit	

## Appendix B

### Equivalence of Depolarizing Channel and Unitary Over-Rotations

Our model includes a decoherence model, the depolarizing channel, as well as operational errors, i.e. unitary over-rotations, and we can show that for a single qubit and certain parameter values both error approaches are equivalent to each other. We will show that the choice of rotation angles will be relevant as well as the possibility to choose a non-equally weighted error basis. Certainly for the circuit of a quantum algorithm we have to distinguish between memory error locations and operational error locations. Memory error locations occur on each qubit and at each timestep, with the time unit determined by the execution time of a gate. Operational error locations only affect a qubit, when an operation is done on that qubit. This approach is similar to an analysis made in [Steane, 2003], where two error rates are used, the gate ( $\gamma$ ) and memory ( $\epsilon$ ) failure rates, and thresholds are determined depending on  $\epsilon/\gamma$ .

For the proof it is necessary and sufficient to show that for both approaches each of the expectation values for X, Y and Z agree. We start with the general case where a single qubit is in an arbitrary state  $|\psi\rangle = \alpha|0\rangle + \beta|1\rangle$  and examine the expectation values  $\langle X \rangle$ ,  $\langle Y \rangle$  and  $\langle Z \rangle$ . For the depolarizing channel we have

$$\begin{aligned}\langle X \rangle &= (1-p) \langle \psi | X | \psi \rangle + \frac{p}{3} (\langle \psi | X^\dagger X X | \psi \rangle + \langle \psi | Y^\dagger X Y | \psi \rangle + \langle \psi | Z^\dagger X Z | \psi \rangle) \\ &= \dots \\ &= (\alpha\beta^* + \alpha^*\beta) \left(1 - \frac{4}{3}p\right),\end{aligned}\tag{B.1}$$

$$\begin{aligned}\langle Y \rangle &= (1-p) \langle \psi | Y | \psi \rangle + \frac{p}{3} (\langle \psi | X^\dagger Y X | \psi \rangle + \langle \psi | Y^\dagger Y Y | \psi \rangle + \langle \psi | Z^\dagger Y Z | \psi \rangle) \\ &= i(\alpha\beta^* - \alpha^*\beta) \left(1 - \frac{4}{3}p\right),\end{aligned}\tag{B.2}$$



APPENDIX B. EQUIVALENCE OF DEPOLARIZING CHANNEL AND UNITARY OVER-ROTATIONS

---

and

$$\begin{aligned}\langle Z \rangle &= (1-p) \langle \psi | Z | \psi \rangle + \frac{p}{3} (\langle \psi | X^\dagger Z X | \psi \rangle + \langle \psi | Y^\dagger Z Y | \psi \rangle + \langle \psi | Z^\dagger Z Z | \psi \rangle) \\ &= (|\alpha|^2 - |\beta|^2) \left(1 - \frac{4}{3}p\right).\end{aligned}\tag{B.3}$$

Next, we determine the expectation values for the unitary over-rotations. We start with the first case, where we use the Euler angle definition, i.e., we decompose an arbitrary rotation  $R_{\vec{n}}$  around an axis  $\vec{n}$  using two rotational axes. In our error model we use plane rotations around the  $y$ -axis  $R_y$  and phase rotations about the  $z$ -axis  $R_z$ :

$$R_{\vec{n}}(\omega) = R_z(\phi') R_y(\theta) R_z(\phi),\tag{B.4}$$

with the matrix representations

$$R_y(\theta) = \begin{pmatrix} \cos \frac{\theta}{2} & -\sin \frac{\theta}{2} \\ \sin \frac{\theta}{2} & \cos \frac{\theta}{2} \end{pmatrix}\tag{B.5}$$

and

$$R_z(\phi) = \begin{pmatrix} 1 & 0 \\ 0 & e^{i\phi} \end{pmatrix}.\tag{B.6}$$

If we assume small over-rotation errors  $\epsilon_\phi$ ,  $\epsilon_\theta$  and  $\epsilon_{\phi'}$ , the expectation value for the operator  $O \in \{X, Y, Z\}$  is given by

$$\begin{aligned}\langle O \rangle &= \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} \langle \psi | R_z^\dagger(\epsilon_\phi) R_y^\dagger(\epsilon_\theta) R_z^\dagger(\epsilon_{\phi'}) O R_z(\epsilon_{\phi'}) R_y(\epsilon_\theta) R_z(\epsilon_\phi) | \psi \rangle \\ &\quad p(\epsilon_\phi) p(\epsilon_\theta) p(\epsilon_{\phi'}) d\epsilon_\phi d\epsilon_\theta d\epsilon_{\phi'},\end{aligned}\tag{B.7}$$

with

$$p(\epsilon) = \frac{1}{\sqrt{2\pi}\sigma} e^{-\epsilon/(2\sigma^2)}\tag{B.8}$$

being the Gauss distribution with standard deviation  $\sigma$ .

Evaluating the integrand is a tedious but straightforward calculation. We get

$$\begin{aligned}\langle X \rangle &= \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} (|\alpha|^2 - |\beta|^2) (\cos \epsilon_{\phi'} \cos \frac{\epsilon_\theta}{2} \sin \frac{\epsilon_\theta}{2}) \\ &\quad + \alpha^* \beta e^{i\phi} (e^{i\phi'} \cos^2 \frac{\epsilon_\theta}{2} - e^{-i\phi'} \sin^2 \frac{\epsilon_\theta}{2}) \\ &\quad + \alpha \beta^* e^{-i\phi} (e^{-i\phi'} \cos^2 \frac{\epsilon_\theta}{2} - e^{i\phi'} \sin^2 \frac{\epsilon_\theta}{2}) \\ &\quad p(\epsilon_\phi) p(\epsilon_\theta) p(\epsilon_{\phi'}) d\epsilon_\phi d\epsilon_\theta d\epsilon_{\phi'},\end{aligned}\tag{B.9}$$

and after some simplifications using trigonometric identities and paying attention to the fact that the integral from minus infinity to plus infinity over an odd function equals zero, we obtain

$$\begin{aligned} \langle X \rangle = & \alpha^* \beta \int_{-\infty}^{\infty} e^{i\epsilon_\phi} p(\epsilon_\phi) d\epsilon_\phi \\ & \left( \int_{-\infty}^{\infty} e^{i\epsilon_{\phi'}} p(\epsilon_{\phi'}) d\epsilon_{\phi'} - 2 \int_{-\infty}^{\infty} \cos \epsilon_{\phi'} p(\epsilon_{\phi'}) d\epsilon_{\phi'} \int_{-\infty}^{\infty} \sin^2 \frac{\epsilon_\theta}{2} p(\epsilon_\theta) d\epsilon_\theta \right) \\ & + \alpha \beta^* \int_{-\infty}^{\infty} e^{-i\epsilon_\phi} p(\epsilon_\phi) d\epsilon_\phi \\ & \left( \int_{-\infty}^{\infty} e^{-i\epsilon_{\phi'}} p(\epsilon_{\phi'}) d\epsilon_{\phi'} - 2 \int_{-\infty}^{\infty} \cos \epsilon_{\phi'} p(\epsilon_{\phi'}) d\epsilon_{\phi'} \int_{-\infty}^{\infty} \sin^2 \frac{\epsilon_\theta}{2} p(\epsilon_\theta) d\epsilon_\theta \right). \end{aligned} \quad (\text{B.10})$$

After evaluating the remaining integrals and simplifying the expression we obtain

$$\langle X \rangle = (\alpha^* \beta + \alpha \beta^*) e^{-\frac{3}{2}\sigma^2}. \quad (\text{B.11})$$

In line with the evaluation of the first expectation value, the computation of the remaining expectation values yield

$$\langle Y \rangle = i(\alpha \beta^* - \alpha^* \beta) e^{-\sigma^2} \quad (\text{B.12})$$

and

$$\langle Z \rangle = (|\alpha|^2 - |\beta|^2) e^{-\frac{1}{2}\sigma^2}. \quad (\text{B.13})$$

A comparison of equations (B.1), (B.2) and (B.3) with equations (B.11), (B.12) and (B.13) shows that, if we drive our error models with a single decoherence parameter  $p$  and a single Gaussian width  $\sigma$ , there is no one-to-one transformation between depolarizing channel errors and operational unitary over-rotations.<sup>1</sup>

Apparently, we have made a non-isotropic choice by using the Euler angles (equation (B.4)) for our rotations, so it is not a surprise to see different transformation rules for different directions. To check if the anisotropy is just attributed to the choice of rotation angles, we repeated the same calculations again, but this time starting with a decomposition of an arbitrary rotation into a product of rotations around the three axes,

$$R_{\vec{n}}(\omega) = R_x(\theta_x) R_y(\theta_y) R_z(\theta_z). \quad (\text{B.17})$$

<sup>1</sup>The fact that we used equation (B.5) instead of equation (2.72), where the rotation angle is twice as large, does not change this conclusion. If we use equation (2.72) we get the results

$$\langle X \rangle = (\alpha^* \beta + \alpha \beta^*) e^{-3\sigma^2}, \quad (\text{B.14})$$

$$\langle Y \rangle = i(\alpha \beta^* - \alpha^* \beta) e^{-\sigma^2} \quad (\text{B.15})$$

$$\langle Z \rangle = (|\alpha|^2 - |\beta|^2) e^{-2\sigma^2}. \quad (\text{B.16})$$

This time the calculation is even more laborious, because we have to evaluate

$$\begin{aligned}
& R_x(\theta_x)R_y(\theta_y)R_z(\theta_z) \\
&= \left( \cos \frac{\theta_x}{2} \cos \frac{\theta_y}{2} \cos \frac{\theta_z}{2} - \sin \frac{\theta_x}{2} \sin \frac{\theta_y}{2} \sin \frac{\theta_z}{2} \right) \mathbb{I} \\
&-i \left( \cos \frac{\theta_y}{2} \cos \frac{\theta_z}{2} \sin \frac{\theta_x}{2} + \cos \frac{\theta_x}{2} \sin \frac{\theta_y}{2} \sin \frac{\theta_z}{2} \right) X \\
&-i \left( \cos \frac{\theta_x}{2} \cos \frac{\theta_z}{2} \sin \frac{\theta_y}{2} - \cos \frac{\theta_y}{2} \sin \frac{\theta_x}{2} \sin \frac{\theta_z}{2} \right) Y \\
&-i \left( \cos \frac{\theta_z}{2} \sin \frac{\theta_x}{2} \sin \frac{\theta_y}{2} + \cos \frac{\theta_x}{2} \cos \frac{\theta_y}{2} \sin \frac{\theta_z}{2} \right) Z,
\end{aligned} \tag{B.18}$$

the product  $R_z^\dagger(\theta_z)R_y^\dagger(\theta_y)R_x^\dagger(\theta_x)OR_x(\theta_x)R_y(\theta_y)R_z(\theta_z)$  with  $O \in \{X, Y, Z\}$  and the expectation value thereof. Again, exploiting trigonometric identities and the knowledge about integrals over odd functions and using the commutation relations of the Pauli matrices eventually yields

$$\begin{aligned}
\langle X \rangle &= (\alpha^* \beta + \alpha \beta^*) \int_{-\infty}^{\infty} (1 - 2 \sin^2 \frac{\theta_y}{2}) p(\theta_y) d\theta_y \int_{-\infty}^{\infty} (1 - 2 \sin^2 \frac{\theta_z}{2}) p(\theta_z) d\theta_z \\
&= (\alpha^* \beta + \alpha \beta^*) e^{-\sigma^2},
\end{aligned} \tag{B.19}$$

$$\begin{aligned}
\langle Y \rangle &= i(\alpha \beta^* - \alpha^* \beta) \int_{-\infty}^{\infty} (1 - 2 \sin^2 \frac{\theta_x}{2}) p(\theta_x) d\theta_x \int_{-\infty}^{\infty} (1 - 2 \sin^2 \frac{\theta_z}{2}) p(\theta_z) d\theta_z \\
&= i(\alpha \beta^* - \alpha^* \beta) e^{-\sigma^2}
\end{aligned} \tag{B.20}$$

and

$$\begin{aligned}
\langle Z \rangle &= (|\alpha|^2 - |\beta|^2) \int_{-\infty}^{\infty} (1 - 2 \sin^2 \frac{\theta_x}{2}) p(\theta_x) d\theta_x \int_{-\infty}^{\infty} (1 - 2 \sin^2 \frac{\theta_y}{2}) p(\theta_y) d\theta_y \\
&= (|\alpha|^2 - |\beta|^2) e^{-\sigma^2}.
\end{aligned} \tag{B.21}$$

Comparing this result with equations (B.1), (B.2) and (B.3) leads to the conclusion that there is a one-to-one correspondence between a depolarizing channel error probability  $p$  and a unitary over-rotation with Gaussian distributed errors with standard deviation  $\sigma$ :

$$1 - \frac{4}{3}p = e^{-\sigma^2}. \tag{B.22}$$

On average a unitary over-rotation where the error is Gauss distributed with standard deviation  $\sigma$  gives the same result as doing a perfect operation after which a discrete error happens with probability  $p = 3/4(1 - e^{-\sigma^2})$ . If we obey this relation, each memory error location

---

can be translated to a unitary over-rotation of the identity operation and each gate error imprecision can be translated to a perfect operation with subsequent discrete error with a well defined probability.

This analysis has also been verified by numerical simulations of a single qubit decohering. A run with only decoherence errors and no operational errors gives exactly the same result as the algorithm run with only unitary over-rotations and no decoherence errors. Both results do agree with the analytically expected expectation values.

We found that under certain conditions the depolarizing channel error model and the unitary over-rotational imprecisions model can be unified. If the discrete error model has an unbiased error basis, like in the depolarizing channel, and the imprecisions are Gaussian distributed in the over-rotations and if we further assume a decomposition of any rotation into rotations around the three canonical axes of the coordinate system, then equation (B.22) gives the relation between both error approaches. However this is not true anymore, when we use Euler rotations, because this introduces an anisotropy and the corresponding approach would require an introduction of a biased discrete error basis, where the probabilities for certain errors are not equal.

The use of both error models in our simulations is not only justified by this observation, but especially the resemblance to the experimental constraints suggests using unitary over-rotations as well in addition to the depolarizing channel that accounts for decoherence (see section 2.2.1).

APPENDIX B. EQUIVALENCE OF DEPOLARIZING CHANNEL AND UNITARY  
OVER-ROTATIONS

---

## Appendix C

# Stabilizer Codes for Quantum Error Correction

Stabilizer codes [Gottesman, 1997] form an important class of quantum codes. They utilize the stabilizer formalism for doing quantum error correction. This section gives a brief summary of the stabilizer formalism and shows how it can be used to deduce quantum error correction codes. The description follows that of [Nielsen and Chuang, 2000].

**Stabilizer formalism** The basic idea of the stabilizer formalism is to describe quantum states by operators that stabilize those states instead of describing the state vector directly. For example the Bell State

$$|\psi\rangle = \frac{|00\rangle + |11\rangle}{\sqrt{2}} \quad (\text{C.1})$$

is an eigenstate of both operators  $X_1X_2$  and  $Z_1Z_2$ :

$$X_1X_2 |\psi\rangle = |\psi\rangle \quad (\text{C.2})$$

and

$$Z_1Z_2 |\psi\rangle = |\psi\rangle. \quad (\text{C.3})$$

In fact,  $|\psi\rangle$  is the unique state which is stabilized by  $X_1X_2$  and  $Z_1Z_2$ .

The power of the stabilizer formalism can be exploited by using group theory. For a single qubit the Pauli Group  $G_1$  is defined as

$$G_1 = \{\pm I, \pm iI, \pm X, \pm iX, \pm Y, \pm iY, \pm Z, \pm iZ\}. \quad (\text{C.4})$$

The Pauli Group of  $n$  qubits is defined to consist of all  $n$ -fold tensor products of Pauli matrices with multiplicative factors  $\pm 1$  and  $\pm i$ .

Let  $S$  be a subgroup of  $G_n$ , then  $V_S$  is defined to be the set of  $n$  qubit states that are stabilized by every element of  $S$ .  $V_S$  is called the vector space stabilized by the stabilizer  $S$ .

Any group  $G$  can be described by its generators  $g_1, \dots, g_l$ , i.e., any element of  $G$  can be written as a product of elements from this list:

$$G = \langle g_1, \dots, g_l \rangle. \quad (\text{C.5})$$

A result from group theory is that a group of size  $|G|$  has a set of at most  $\log(|G|)$  generators, allowing a compact description of the group. Another important fact is that a vector is stabilized by a group  $S$  if and only if it is stabilized by the generators of the group.  $S$  stabilizes a non-trivial vector space  $V_S$  if and only if the elements of  $S$  commute and  $-I$  is not an element of  $S$ .<sup>1</sup>

**Unitary gates in the stabilizer formalism** The stabilizer formalism is not only useful for the description of vector spaces, but also for describing the dynamics of those. Applying a unitary operation  $U$  to a vector  $|\psi\rangle$  in the vector space  $V_S$  stabilized by the group  $S$  for any element  $g$  of  $S$  gives:

$$U|\psi\rangle = Ug|\psi\rangle = UgU^\dagger U|\psi\rangle, \quad (\text{C.6})$$

so the new state  $U|\psi\rangle$  is stabilized by  $UgU^\dagger$ . In consequence the vector space  $UV_S$  is stabilized by the group  $USU^\dagger = \{UgU^\dagger | g \in S\}$  and if  $S$  has generators  $g_1, \dots, g_l$ , then  $Ug_1U^\dagger, \dots, Ug_lU^\dagger$  generate  $USU^\dagger$ .

To illustrate the advantage of this approach, let's look at the following example: The Hadamard gate  $H$  is applied to the state stabilized by  $Z$ , which is the state  $|0\rangle$ . It follows that  $HZH^\dagger = X$ , so the resulting state is stabilized by  $X$ , which is the state  $2^{-1/2}(|0\rangle + |1\rangle)$ . For a single qubit the advantage is not obvious, but now think of a  $n$ -qubit state with stabilizer  $\langle Z_1, Z_2, \dots, Z_n \rangle$ . This is just the state  $|0\rangle^{\otimes n}$ . After applying a Hadamard gate to each of the qubits, the state has the stabilizer  $\langle X_1, X_2, \dots, X_n \rangle$ . It is the state of equal superposition of all computational basis states. The remarkable result is that the state is fully specified by  $n$  generators, instead of using  $2^n$  amplitudes to fully describe the state.

The stabilizer formalism is not constrained to the Pauli group  $G_n$ , but the description also holds true for the Clifford group (also called normalizer group), whose generators are the Hadamard operator, the CNOT and the single qubit phase gate [Gottesman, 1997]. Table C.1 summarizes the transformation properties of elements of the Pauli group under conjugation. Products of operators can be derived from the results of this table.

It turns out that while many interesting gates lie within this group, most quantum gates are outside, especially the  $\pi/8$ -gate (figure 2.2) and the Toffoli gate (appendix A), which are of particular interest for building a universal set of quantum gates (see section 2.1.1). The

---

<sup>1</sup>The necessity can be seen easily; for the proof that these two conditions are also sufficient, refer to [Nielsen and Chuang, 2000].

---

Operation	Input	Output
CNOT	$X_1$	$X_1X_2$
	$X_2$	$X_2$
	$Z_1$	$Z_1$
	$Z_2$	$Z_1Z_2$
H	X	Z
	Z	X
S	X	Y
	Z	Z
X	X	X
	Z	-Z
Y	X	-X
	Z	-Z
Z	X	-X
	Z	Z

**Table C.1** – Transformation properties of Pauli group elements under conjugation.

Generator	Operator
$g_1$	$ZZI$
$g_2$	$I ZZ$

**Table C.2** – Generators of the 3-qubit bit flip code using stabilizer formalism.

$g_1$	$g_2$	error type	action
+1	+1	no error	no action
+1	-1	bit 3 flipped	flip bit 3
-1	+1	bit 1 flipped	flip bit 1
-1	-1	bit 2 flipped	flip bit 2

**Table C.3** – 3-qubit bit flip code in the stabilizer description. Measuring the stabilizers leads to a specific recovery operation.

stabilizer formalism does *not* provide an efficient scheme for general quantum computation, as for universal quantum computation gates are needed, which can move elements out of the Pauli and Clifford group, like the  $\pi/8$ -gate.

Nevertheless, for stabilizer quantum codes, the encoding, error-detection, recovery and decoding can be realized using only gates from this group, so that the use of the stabilizer formalism is well suited for the analysis of such codes.

**3-qubit bit flip code using stabilizer formalism** The 3-qubit bit flip code can be described by the stabilizer generators in table C.2. Measuring the stabilizers results in the actions for the correction of an error given by table C.3.



---

APPENDIX C. STABILIZER CODES FOR QUANTUM ERROR CORRECTION

---

Name	Operator
$g_1$	ZZIIIIIII
$g_2$	IZZIIIIII
$g_3$	IIIZZIIII
$g_4$	IIIIZZIII
$g_5$	IIIIIZZZI
$g_6$	IIIIIIIZZ
$g_7$	XXXXXXIII
$g_8$	IIIXXXXXX
$Z$	XXXXXXXXX
$X$	ZZZZZZZZZ

**Table C.4** – Generators of the stabilizer and logical Z and logical X operations for Shor’s 9-qubit quantum error correction code. The entries represent tensor products on the respective qubits. Note that the logical operations are just the reverse of what one would naively expect.

Generator	Operator
$g_1$	IIIXXXX
$g_2$	IXXIIXX
$g_3$	XIXIXIX
$g_4$	IIIZZZZ
$g_5$	IZZIIIZZ
$g_6$	ZIZIZIZ

**Table C.5** – Generators of the stabilizer for Steane’s 7-qubit quantum error correction code. The entries represent tensor products on the respective qubits.

**9-qubit code using stabilizer formalism** Shor’s 9-qubit code can also be represented in the stabilizer formalism, yielding the generators and logical operators given in table C.4.

**7-qubit code using stabilizer formalism** Steane’s 7-qubit code can be described in a very compact way within the stabilizer formalism (compared to the description in terms of state vectors (equations (2.113) and (2.114))). The generators are given in table C.5. Notice the similarity to the Hamming  $[7, 4, 3]$  code with the parity check matrix given by equation (2.110).

**5-qubit code using stabilizer formalism** The 5-qubit code [Laflamme et al., 1996; Di-Vincenzo and Shor, 1996] is the smallest code that can correct any error on a single qubit. The generators of the stabilizer are given in table C.6. Although the 5-qubit code is the smallest one to protect against any single error, Steane’s 7-qubit code has some properties that makes it much easier to handle, especially when going to fault-tolerant quantum error correction (section 2.3.3).

---

Name	Operator
$g_1$	XZZXI
$g_2$	IXZZX
$g_3$	XIXZZ
$g_4$	ZXIXZ
$\bar{Z}$	ZZZZZ
$\bar{X}$	XXXXX

**Table C.6** – Generators of the stabilizer and logical Z and logical X operation for the 5-qubit quantum error correction code. The entries represent tensor products on the respective qubits. Note that the latter generators are obtained by right shifting the first one.



# Appendix D

## Listings

**Listing D.1** – *Determination of communication partners*

```
stride_tasks=1<<(K-M);

for(i=0; i<L; i+=2)
{
    j = i & stride_tasks;
    task_i = i-j+j/stride_tasks;
    task_j = task_i + stride_tasks;

    if (myrank == task_j)
    {
        /* send odd entries to task_i and get even entries
           from task_i */
        /* calculate even entries */
        /* send back even entries and get back odd entries
           */
    }
    if (myrank == task_i)
    {
        /* send even entries to task_j and get odd entries
           from task_j */
        /* calculate odd entries */
        /* send back odd entries and get back even entries
           */
    }
}
```

**Listing D.2** – *CNOT gate in the case  $M > T$  and  $C > T$ , so that no communication between different MPI tasks is necessary. The explicit distinction between  $T = 0$  and  $T \neq 0$  is necessary due to the even/odd-splitting scheme.*

```
/* Executing loop with parameters set: */
if (T != 0) /* common case */
{
    for (k=kstart; k<=kmax; k+=2*j1)
    {
        for (l=0; l<=lmax; l+=2*i1)
        {
            for (m=0; m<=(i1-1)/evenodd; m++)
            {
                i=(k+1)/evenodd+m;
                j=i+i1/evenodd;

                r0=psi_Ra[i];
                r1=psi_Ia[i];
                r2=psi_Ra[j];
                r3=psi_Ia[j];
                psi_Ra[i]=r2;
                psi_Ia[i]=r3;
                psi_Ra[j]=r0;
                psi_Ia[j]=r1;

                r0=psi_Rb[i];
                r1=psi_Ib[i];
                r2=psi_Rb[j];
                r3=psi_Ib[j];
                psi_Rb[i]=r2;
                psi_Ib[i]=r3;
                psi_Rb[j]=r0;
                psi_Ib[j]=r1;
            }
        }
    }
}
else /* (T == 0) --> even/odd exchange */
{
    for (k=kstart; k<=kmax; k+=2*j1)
    {
        for (l=0; l<=lmax; l+=2) /* 2*i1=2*I=2 */

```

---

```

    {
        i=(k+1)/evenodd;

        r0=psi_Ra[i];
        r1=psi_Ia[i];
        r2=psi_Rb[i];
        r3=psi_Ib[i];
        psi_Ra[i]=r2;
        psi_Ia[i]=r3;
        psi_Rb[i]=r0;
        psi_Ib[i]=r1;
    }
}

```

**Listing D.3** – *CNOT gate in the case  $M > T$  and  $C < T$ . No communication between different MPI tasks is necessary. The explicit distinction between  $T = 0$  and  $T \neq 0$  is necessary due to the even/odd-splitting scheme.*

```

if (control_bit != 0)
{
    for (k=j1; k<=j1+1<<M-2*i1; k+=2*i1)
    {
        for (l=0; l<=i1-2*j1; l+=2*j1)
        {
            for (m=0; m<=(j1-1)/evenodd; m++)
            {
                i=(k+1)/evenodd+m;
                j=i+i1/evenodd;

                r0=psi_Ra[i];
                r1=psi_Ia[i];
                r2=psi_Ra[j];
                r3=psi_Ia[j];
                psi_Ra[i]=r2;
                psi_Ia[i]=r3;
                psi_Ra[j]=r0;
                psi_Ia[j]=r1;

                r0=psi_Rb[i];
                r1=psi_Ib[i];
                r2=psi_Rb[j];

```

```

        r3=psi_Ib[j];
        psi_Rb[i]=r2;
        psi_Ib[i]=r3;
        psi_Rb[j]=r0;
        psi_Ib[j]=r1;
    }
}
}
}
}
else /* (control_bit == 0) only odd-elements involved */
{
    for(k=j1; k<=j1+NSTATESPERTASK-2*i1; k+=2*i1)
    {
        for(l=0; l<=i1-2; l+=2) /* 2*j1=2*l=2 */
        {
            for(m=0; m<=(j1-1)/evenodd; m++)
            {
                i=(k+1)/evenodd+m;
                j=i+i1/evenodd;

                r0=psi_Rb[i];
                r1=psi_Ib[i];
                r2=psi_Rb[j];
                r3=psi_Ib[j];
                psi_Rb[i]=r2;
                psi_Ib[i]=r3;
                psi_Rb[j]=r0;
                psi_Ib[j]=r1;
            }
        }
    }
}
}

```

**Listing D.4** – *CNOT gate in the case  $T \leq M$  and  $C > T$ . Communication between different MPI tasks is unavoidable. The explicit distinction between  $T = 0$  and  $T \neq 0$  is not necessary, because an operation on qubit 0 can always be done locally*

```

if( (myrank/sw)%2 == 1)
{
    if( (myrank/sw2)%2 == 0) /* first communication partner */
    {
        /* Send odd part to myrank+sw2; */
        /* Receive even part from myrank+sw2: */
    }
}

```

---

```

    /* Exchange amplitudes */
    /* backward communication */
}
else /* second communication partner */
{
    /* Send even part to myrank-sw2; */
    /* receive odd part from myrank-sw2: */
    /* Exchange amplitudes */
    /* backward communication */
}
}

```

**Listing D.5** – CNOT gate in the case  $T \leq M$  and  $C < T$ . Communication between different MPI tasks is unavoidable.

```

if (sw == 0) /* M > C */
{
    kstart=j1;
    kmax=j1; /* mainly 1 iteration for k-loop */
    lmax=1<<M-2*j1;
    mmax=j1-1;
}
else /* C >= M */
{
    if ( (myrank/sw)%2 == 0)
        kstart=j1; /* mainly sth. greater than kmax=0 */
    else
        kstart=0;

    kmax=0;
    lmax=0;
    mmax=1<<M-1;
}

/* Executing loop with parameters set: */
for (k=kstart; k<=kmax; k+=2*i1) /* Actually, step doesn't
    matter */
{
    /* if there is anything to do, do communication part here:
       */

    if (control_bit != 0) /* common case */

```



```
{
  /* otherwise only odd states involved -> less commun. */
  if( (myrank/sw2)%2 == 0) /* first communication partner
    */
  {
    /* Send odd part to myrank+sw2; */
    /* Receive even part from myrank+sw2: */

    for( l=0; l<=lmax; l+=2*j1 )
    {
      for( m=0; m<=mmax/evenodd; m++)
      {
        i=(k+1)/evenodd+m;
        /* swap amplitudes */
      }
    }

    /* backward communication: */
  }
  else /* second communication partner */
  {
    /* Send even part to myrank-sw2; */
    /* receive odd part from myrank-sw2: */

    for( l=0; l<=lmax; l+=2*j1 )
    {
      for( m=0; m<=mmax/evenodd; m++)
      {
        i=(k+1)/evenodd+m;
        /* swap amplitudes */
      }
    }
    /* backward communication: */
  }
}
else /* (control_bit==0), only odd states, less commun. */
{
  if( (myrank/sw2)%2 == 0 )
  {
    /* Send odd part to myrank+sw2 */
    /* nothing to do here */
  }
}
```

---

```

        /* backward communication: */
    }
    else
    {
        /* Receive odd part from myrank-sw2: */

        for (l=0; l<=lmax; l+=2*j1)
        {
            for (m=0; m<=mmax/evenodd; m++)
            {
                i=(k+1)/evenodd+m;
                /* swap amplitudes */
            }
        }
        /* backward communication: */
    }
}
}
}

```

## APPENDIX D. LISTINGS

---

## Bibliography

- Aharonov, D. and Ben-Or, M. (1996). Fault tolerant quantum computation with constant error. quant-ph/9611025v2.
- Aliferis, P., Gottesman, D., and Preskill, J. (2005). Quantum accuracy threshold for concatenated distance-3 codes. *Quantum Information & Computation*, 6:97–165.
- Barenco, A., Bennett, C., Cleve, R., DiVincenzo, D., Margolus, N., Shor, P., Sleator, T., Smolin, J., and Weinfurter, H. (1995). Elementary gates for quantum computation. *Physical Review A*, 52:3457–3467.
- Barenco, A., Ekert, A., Suominen, K., and Torma, P. (1996). Approximate quantum Fourier transform and decoherence. *Physical Review A*, 54:139–146.
- Benhelm, J. (2008). Experimental quantum-information processing with  $^{43}\text{Ca}^+$  ions. *Physical Review A*, 77(6):062306.
- Benioff, P. (1982). Quantum-mechanical hamiltonian models of turing-machines. *Journal Of Statistical Physics*, 29:515–546.
- Bennett, C., Brassard, G., Crepeau, C., Jozsa, R., Peres, A., and Wootters, W. (1993). Teleporting an unknown quantum state via dual classical and Einstein-Podolsky-Rosen channels. *Physical Review Letters*, 70:1895–1899.
- Byrnes, T. and Yamamoto, Y. (2006). Simulating lattice gauge theories on a quantum computer. *Physical Review A*, 73:022328.
- Calderbank, A. and Shor, P. (1996). Good quantum error-correcting codes exist. *Physical Review A*, 54:1098–1105.
- Chapman, B., Jost, G., and Pas, R. v. d. (2007). *Using OpenMP: Portable Shared Memory Parallel Programming*. The MIT Press.
- Childs, A. and Chuang, I. (2001). Universal quantum computation with two-level trapped ions. *Physical Review A*, 63(1):012306.
- Cirac, J. and Zoller, P. (1995). Quantum computations with cold trapped ions. *Physical Review Letters*, 74(20):4091–4094.
- Cirac, J., Zoller, P., Kimble, H., and Mabuchi, H. (1997). Quantum state transfer and entanglement distribution among distant nodes in a quantum network. *Physical Review Letters*, 78:3221–3224.

## BIBLIOGRAPHY

---

- Cohen-Tannoudji, C., Diu, B., and Laloe, F. (1978). *Quantum Mechanics*. John Wiley & Sons.
- De Raedt, H. (1987). Product formula algorithms for solving the time dependent Schrödinger equation. *Computer Physics Reports*, 7:1–72.
- De Raedt, K., Michielsen, K., De Raedt, H., Trieu, B., Arnold, G., Richter, M., Lippert, T., Watanabe, H., and Ito, N. (2007). Massively parallel quantum computer simulator. *Computer Physics Communications*, 176(2):121–136.
- Deutsch, D. (1985). Quantum-theory, the Church-Turing principle and the universal quantum computer. *Proceedings Of The Royal Society Of London Series A-Mathematical Physical And Engineering Sciences*, 400:97–117.
- Deutsch, D. (1989). Quantum computational networks. *Proceedings of the Royal Society of London, Series A: Mathematical, Physical and Engineering Sciences*, 425:73–90.
- DiVincenzo, D. (1995). Two-bit gates are universal for quantum computation. *Physical Review A*, 51:1015–1022.
- DiVincenzo, D. (2000). The physical implementation of quantum computation. *Fortschritte der Physik*, 48:771–783.
- DiVincenzo, D. and Shor, P. (1996). Fault-tolerant error correction with efficient quantum codes. *Physical Review Letters*, 77:3260–3263.
- Earnshaw, S. (1842). On the nature of the molecular forces which regulate the constitution of the luminiferous ether. *Transactions of the Cambridge Philosophical Society*, 7:97–112.
- Farhi, E., Goldstone, J., Gutmann, S., and Sipser, M. (2000). Quantum computation by adiabatic evolution. quant-ph/0001106.
- Feynman, R. (1982). Simulating physics with computers. *International Journal of Theoretical Physics*, 21:467–488.
- Fowler, A. and Hollenberg, L. (2004). Scalability of Shor’s algorithm with a limited set of rotation gates. *Physical Review A*, 70(3):032329.
- Fowler, A. G. (2005). *Towards Large-Scale Quantum Computation*. PhD thesis, School of Physics, University of Melbourne.
- Frigo, M. and Johnson, S. G. (2005). The design and implementation of FFTW3. *Proceedings of the IEEE*, 93(2):216–231. Special issue on “Program Generation, Optimization, and Platform Adaptation”.
- Galassi, M., Davies, J., Theiler, J., Gough, B., Jungman, G., Booth, M., and Rossi, F., editors (2006). *GNU Scientific Library Reference Manual*. Network Theory Limited, 2nd revised edition.
- Gershenfeld, N. and Chuang, I. (1997). Bulk spin-resonance quantum computation. *Science*, 275(5298):350–356.

- Gottesman, D. (1997). *Stabilizer Codes and Quantum Error Correction*. PhD thesis, California Institute of Technology.
- Gottesman, D. (1998). A theory of fault-tolerant quantum computation. *Physical Review A*, 57:127.
- Gropp, W., Lusk, E., and Skjellum, A. (1994). *Using MPI: Portable Parallel Programming with the Message-Passing Interface*. The MIT Press.
- Grover, L. K. (1996). A fast quantum mechanical algorithm for database search. In *Proceedings of 28th Annual ACM Symposium on Theory of Computing*, pages 212–219. ACM Press.
- Gulde, S. (2003). *Experimental Realization of Quantum Gates and the Deutsch-Jozsa Algorithm with Trapped  $^{40}\text{Ca}^+$  Ions*. PhD thesis, Leopold-Franzens-Universität Innsbruck.
- Häffner, H., Hänsel, W., Roos, C., Benhelm, J., Chek-al kar, D., Chwalla, M., Korber, T., Rapol, U., Riebe, M., Schmidt, P., Becher, C., Guhne, O., Dür, W., and Blatt, R. (2005). Scalable multiparticle entanglement of trapped ions. *Nature*, 438(7068):643–646.
- Häffner, H., Roos, C., and Blatt, R. (2008). Quantum computing with trapped ions. *Physics Reports*, 469(4):155–203.
- Hill, C. R. and Viamontes, G. F. (2008). Operator imprecision and scaling of Shor’s algorithm. quant-ph/08043076.
- Hughes, R. (2004). A quantum information science and technology roadmap - part 1: Quantum computation. Technical report, Los Alamos National Laboratory.
- James, D. (1998). Quantum dynamics of cold trapped ions with application to quantum computation. *Applied Physics B: Lasers and Optics*, 66:181–190.
- Jaynes, E. T. and Cummings, F. W. (1963). Comparison of quantum and semiclassical radiation theories with application to the beam maser. *Proceedings of the IEEE*, 51(1):89–109.
- Johnston, F. and King-Smith, B. (2005). IBM pSeries High Performance Switch. Performance white paper, IBM Systems and Technology Group, Poughkeepsie, New York.
- Khaneja, N., Reiss, T., Kehlet, C., Schulte-herbruggen, T., and Glaser, S. (2005). Optimal control of coupled spin dynamics: Design of NMR pulse sequences by gradient ascent algorithms. *Journal of Magnetic Resonance*, 172(2):296–305.
- Kitaev (1997a). Quantum error correction with imperfect gates. In Hirota O, Holevo AS, C. C., editor, *Quantum Communication, Computing, and Measurement*, pages 181–188. Plenum Press.
- Kitaev, A. (1997b). Quantum computations: Algorithms and error correction. *Russian Mathematical Surveys*, 52:1191–1249.
- Kitaev, A. (2003). Fault-tolerant quantum computation by anyons. *Annals of Physics*, 303:2–30.

## BIBLIOGRAPHY

---

- Knill, E. (2005). Quantum computing with realistically noisy devices. *Nature*, 434(7029):39–44.
- Knill, E., Laflamme, R., and Zurek, W. (1996). Threshold accuracy for quantum computation. quant-ph/9610011.
- Knill, E., Laflamme, R., and Zurek, W. (1998a). Resilient quantum computation. *Science*, 279(5349):342–345.
- Knill, E., Laflamme, R., and Zurek, W. (1998b). Resilient quantum computation: Error models and thresholds. *Proceedings of the Royal Society of London, Series A: Mathematical, Physical and Engineering Sciences*, 454:365–384.
- Knill, E., Leibfried, D., Reichle, R., Britton, J., Blakestad, R., Jost, J., Langer, C., Ozeri, R., Seidelin, S., and Wineland, D. (2008). Randomized benchmarking of quantum gates. *Physical Review A*, 77(1):012307.
- Laflamme, R., Miquel, C., Paz, J., and Zurek, W. (1996). Perfect quantum error correcting code. *Physical Review Letters*, 77:198–201.
- Landauer, R. (1995). Is quantum mechanics useful? *Philosophical Transactions of the Royal Society of London, Series A: Mathematical, Physical and Engineering Sciences*, 353:367–376.
- Leibfried, D., Blatt, R., Monroe, C., and Wineland, D. (2003a). Quantum dynamics of single trapped ions. *Reviews of Modern Physics*, 75:281–324.
- Leibfried, D., Demarco, B., Meyer, V., Lucas, D., Barrett, M., Britton, J., Itano, W., Jelenkovic, B., Langer, C., Rosenband, T., and Wineland, D. (2003b). Experimental demonstration of a robust, high-fidelity geometric two ion-qubit phase gate. *Nature*, 422(6930):412–415.
- Leibfried, D., Demarco, B., Meyer, V., Rowe, M., Ben-kish, A., Britton, J., Itano, W., Jelenkovic, B., Langer, C., Rosenband, T., and Wineland, D. (2002). Trapped-ion quantum simulator: Experimental application to nonlinear interferometers. *Physical Review Letters*, 89:247901.
- Lloyd, S. (1996). Universal quantum simulators. *Science*, 273:1073–1078.
- Loss, D. and DiVincenzo, D. (1998). Quantum computation with quantum dots. *Physical Review A*, 57:120–126.
- MacWilliams, F. J. and Sloane, N. J. A. (1977). *The theory of error-correcting codes*. North Holland.
- Monroe, C., Meekhof, D., King, B., Itano, W., and Wineland, D. (1995). Demonstration of a fundamental quantum logic gate. *Physical Review Letters*, 75:4714–4717.
- Monroe, C., Meekhof, D., King, B., and Wineland, D. (1996). A "Schrödinger cat" superposition state of an atom. *Science*, 272:1131–1136.

- Munro, W., Milburn, G., and Sanders, B. (2000). Entangled coherent-state qubits in an ion trap. *Physical Review A*, 62:052108.
- Nägerl, H., Bechter, W., Eschner, J., Schmidt-Kaler, F., and Blatt, R. (1998). Ion strings for quantum gates. *Applied Physics B: Lasers and Optics*, 66:603–608.
- Nielsen, M. and Chuang, I. (2000). *Quantum Computation and Quantum Information*. Cambridge University Press.
- Niwa, J. (2002). General-purpose parallel simulator for quantum computing. *Physical Review A*, 66(6):062317.
- Niwa, J., Matsumoto, K., and Imai, H. (2002). Simulating the effects of quantum error-correction schemes. quant-ph/0211071.
- Oi, D., Devitt, S., and Hollenberg, L. (2006). Scalable error correction in distributed ion trap computers. *Physical Review A*, 74(5):052313.
- Paul, W. (1990). Electromagnetic traps for charged and neutral particles. *Reviews of Modern Physics*, 62:531–540.
- Peschina, M. (2008). Correcting erroneous quantum algorithms. In Bolten, M., editor, *Beiträge zum Wissenschaftlichen Rechnen - Ergebnisse des Gaststudentenprogramms 2008 des John von Neumann-Instituts für Computing*, pages 81–92. Forschungszentrum Jülich.
- Petta, J., Johnson, A., Taylor, J., Laird, E., Yacoby, A., Lukin, M., Marcus, C., Hanson, M., and Gossard, A. (2005). Coherent manipulation of coupled electron spins in semiconductor quantum dots. *Science*, 309(5744):2180–2184.
- Pomplun, N. (2005). Simulation eines Quantencomputers auf einem Höchstleistungsrechner und Analyse der Fehlertoleranz. Master’s thesis, Technische Universität Berlin, Forschungszentrum Jülich GmbH.
- Porras, D. and Cirac, J. (2004). Effective quantum spin systems with trapped ions. *Physical Review Letters*, 92:207901.
- Preskill, J. (1998). Reliable quantum computers. *Proceedings of the Royal Society of London, Series A: Mathematical, Physical and Engineering Sciences*, 454:385–410.
- Raimond, J., Brune, M., and Haroche, S. (2001). Colloquium: Manipulating quantum entanglement with atoms and photons in a cavity. *Reviews of Modern Physics*, 73:565–582.
- Raussendorf, R. and Briegel, H. (2001). A one-way quantum computer. *Physical Review Letters*, 86(22):5188–5191.
- Reichardt, B. W. (2006). *Error-detection-based quantum fault tolerance against discrete Pauli noise*. PhD thesis, University of California, Berkeley.
- Salas, P. (2008). Noise effect on Grover algorithm. *European Physical Journal D*, 46(2):365–373.



## BIBLIOGRAPHY

---

- Sasura, M. and Buzek, V. (2002). Cold trapped ions as quantum information processors. *Journal of Modern Optics*, 49(10):1593–1647.
- Schmidt-Kaler, F., Häffner, H., Gulde, S., Riebe, M., Lancaster, G., Deuschle, T., Becher, C., Hansel, W., Eschner, J., Roos, C., and Blatt, R. (2003a). How to realize a universal quantum gate with trapped ions. *Applied Physics B: Lasers and Optics*, 77(8):789–796.
- Schmidt-Kaler, F., Häffner, H., Riebe, M., Gulde, S., Lancaster, G., Deuschle, T., Becher, C., Roos, C., Eschner, J., and Blatt, R. (2003b). Realization of the Cirac-Zoller controlled-NOT quantum gate. *Nature*, 422(6930):408–411.
- Schrödinger, E. (1935). Discussion of probability relations between separated systems. *Mathematical Proceedings of the Cambridge Philosophical Society*, 31(04):555–563.
- Shor, P. (1994). Algorithms for quantum computation: Discrete logarithms and factoring. In Goldwasser, S., editor, *Proceedings, 35th Annual Symposium on Foundations of Computer Science*, Annual Symposium on Foundations of Computer Science, pages 124–134. IEEE Computer Society Press.
- Shor, P. (1995). Scheme for reducing decoherence in quantum computer memory. *Physical Review A*, 52:R2493–R2496.
- Shor, P. (1996). Fault-tolerant quantum computation. In *37th Annual Symposium on Foundations of Computer Science, Proceedings*, Annual Symposium on Foundations of Computer Science, pages 56–65, AT&T Bell Labs Research, Murray Hill, NJ 07974, USA. IEEE Computer Society Press.
- Steane, A. (1996a). Error correcting codes in quantum theory. *Physical Review Letters*, 77:793–797.
- Steane, A. (1996b). Multiple-particle interference and quantum error correction. *Proceedings of the Royal Society of London, Series A: Mathematical, Physical and Engineering Sciences*, 452:2551–2577.
- Steane, A. (1997). Active stabilization, quantum computation, and quantum state synthesis. *Physical Review Letters*, 78(11):2252–2255.
- Steane, A. (2003). Overhead and noise threshold of fault-tolerant quantum error correction. *Physical Review A*, 68:042322.
- Steane, A., Roos, C., Stevens, D., Mundt, A., Leibfried, D., Schmidt-Kaler, F., and Blatt, R. (2000). Speed of ion-trap quantum-information processors. *Physical Review A*, 62:042305.
- Unruh, W. (1995). Maintaining coherence in quantum computers. *Physical Review A*, 51:992–997.
- Wineland, D., Monroe, C., Itano, W., Leibfried, D., King, B., and Meekhof, D. (1998). Experimental issues in coherent quantum-state manipulation of trapped atomic ions. *Journal of Research of the National Institute of Standards and Technology*, 103:259–328.

## BIBLIOGRAPHY

---

- Wootters, W. and Zurek, W. (1982). A single quantum cannot be cloned. *Nature*, 299:802–803.
- Zalka, C. (1996). Threshold estimate for fault tolerant quantum computation. quant-ph/9612028v2.



**1. Three-dimensional modelling of soil-plant interactions: Consistent coupling of soil and plant root systems**

by T. Schröder (2009), VIII, 72 pages

ISBN: 978-3-89336-576-0

URN: urn:nbn:de: 0001-00505

**2. Large-Scale Simulations of Error-Prone Quantum Computation Devices**

by D. B. Trieu (2009), VI, 173 pages

ISBN: 978-3-89336-601-9

URN: urn:nbn:de: 0001-00552

The theoretical concepts of quantum computation in the idealized and undisturbed case are well understood. However, in practice, all quantum computation devices do suffer from decoherence effects as well as from operational imprecisions. This work assesses the power of error-prone quantum computation devices using large-scale numerical simulations on parallel supercomputers. We present the *Jülich Massively Parallel Ideal Quantum Computer Simulator (JUMPIQCS)*, that simulates a generic quantum computer on gate level. The robustness of various algorithms in the presence of noise has been analyzed. The simulation results show that for large system sizes and long computations it is imperative to actively correct errors by means of fault-tolerant quantum error correction. Fault-tolerant methods require the single qubit error rate to be below a certain threshold. We determined this threshold numerically for Steane's 7-qubit code. Using the depolarizing channel as the source of decoherence, we find a threshold error rate of  $(5.2 \pm 0.2) \cdot 10^{-6}$ . For Gaussian distributed operational over-rotations the threshold lies at a standard deviation of  $0.0431 \pm 0.0002$ . We can conclude that quantum error correction is especially well suited for the correction of operational imprecisions and systematic over-rotations.

For realistic simulations of specific quantum computation devices we extend the generic model to dynamic simulations of realistic hardware models. We focus on today's most advanced technology, i.e. ion trap quantum computation. We developed the *Dynamic Quantum Computer Simulator for Ion Traps (DyQCSI)*. Starting from a microscopic Hamiltonian, it does not rely on approximations that are usually necessary for an analytical approach. We show that the effects due to these approximations are significant. We present several ways for the visualization of the state of the system during its time evolution and demonstrated the benefit of the simulation approach for parameter optimizations.

This publication was written at the Jülich Supercomputing Centre (JSC) which is an integral part of the Institute for Advanced Simulation (IAS). The IAS combines the Jülich simulation sciences and the supercomputer facility in one organizational unit. It includes those parts of the scientific institutes at Forschungszentrum Jülich which use simulation on supercomputers as their main research methodology.